Mellanox WinOF VPI User Manual

Rev 4.80.50000

www.mellanox.com

Mellanox Technologies
350 Oakmead Parkway Suite 100
Sunnyvale, CA 94085
U.S.A.
www.mellanox.com
Tel: (408) 970-3400
Fax: (408) 970-3403

Mellanox Technologies, Ltd.
Beit Mellanox
PO Box 586 Yokneam 20692
Israel
www.mellanox.com
Tel: +972 (0)74 723 7200
Fax: +972 (0)4 959 3245

Table of Contents

List of Tables

Revision History

Table 1 - Revision History

Document Revision	Date	Changes
Rev 4.80.50000	November 27, 2014	Updated section: • Section 3.5.4, "Single Root I/O Virtualization (SR-IOV)", on page 71 and its subsections
	August 30, 2014	Added the following sections: • Section 3.8.4.1.4, "Propriety RDMA Activity", on page 126 • Section 3.6.9, "MLX BUS Registry Keys", on page 106 • Section 4.1, "Snapshot Tool", on page 128 • Section 3.3.1.5, "Multiple Interfaces over non-default PKeys Support", on page 63 • Section 5.5, "Virtualization Related Troubleshooting", on page 143 Updated the following sections: • Section 4.4, "InfiniBand Fabric Diagnostic Utilities", on page 131 • Section 4.5, "Fabric Performance Utilities", on page 134 • Section 4.2, "part_man - Virtual IPoIB Port Creation Utility", on page 128

Table 1 - Revision History

Document Revision	Date	Changes
Rev 4.70	May 4, 2014	Updated the following sections: • Section 1.2, "WinOF Set of Documentation", on page 14 • Section 2.7, "Firmware Upgrade", on page 25 • Section 3.5.4.4.2, "Enabling SR-IOV in Mellanox WinOF Package (Ethernet SR-IOV Only)", on page 80 • Section 3.7.2.1, "Verifying Network Adapter Configuration", on page 77 • Section 5.3, "Ethernet Related Troubleshooting", on page 140 Added the following sections: • Section 2.4, "Installing Mellanox WinOF Driver", on page 18 • Section 2.6, "Uninstalling Mellanox WinOF Driver", on page 25 • Section 3.5.3.3, "Removing NVGRE configuration", on page 70 • Section 3.5.4, "Single Root I/O Virtualization (SR-IOV)", on page 71 • Section 3.5.1, "Virtual Ethernet Adapter", on page 66 • Section 3.5.5, "IPoIB SR-IOV over KVM", on page 84
		• Section 3.1.11, "Lossless TCP", on page 56 • Section 2.9, "Booting Windows from an iSCSI Target", on page 26 • Section 1, "Running Windows as VM over ESX with Mellanox HCAs", on page 134 • Section C, "Registry Keys", on page 152 Removed the following sections: • Documentation
Rev 4.60	February 13, 2014	Updated the following sections: • Section 3.5.2, "Hyper-V with VMQ", on page 67 • Section 3.5.8.1, "Enabling/Disabling NVGRE Offloading", on page 50 Added the following sections: • Section 3.5.3.2, "Verifying the Encapsulation of the Traffic", on page 70 • Section 3.5.1, "Virtual Ethernet Adapter", on page 66

Table 1 - Revision History

Document Revision	Date	Changes
	December 30, 2013	Updated the following sections: • Section 3.1.4.2.2, "Configuring Windows Host", on page 37 - Updated the example in Step 5 • Section 7.1.4.1, "Performance Tuning Tool Application", on page 83 - Updated the Options table • Section 7.2, "Application Specific Optimization and Tuning", on page 87 - Removed the "Bus-master DMA Operations" • Section 5, "OpenSM - Subnet Manager", on page 80 - Added an option of how to register OpemSM via the PowerShell • Section 3.5.3.1.1, "Configuring the NVGRE using PowerShell", on page 69 Added the following sections: • Section 3.1.8, "Configuring Quality of Service (QoS)", on page 47 • Appendix A: "NVGRE Configuration Scripts Examples," on page 147
Rev 4.55	December 15, 2013	Updated the following sections: • Section 3.1.5, "Load Balancing, Fail-Over (LBFO) and VLAN", on page 40 • Section 3.5.3.1.1, "Configuring the NVGRE using PowerShell", on page 69
	November 07, 2013	Updated the following sections: • Section 3.1.4.2.2, "Configuring Windows Host", on page 37 • Section 11.4.19.1, "NTttcp Synopsis", on page 168
	October 03, 2013	Added support for Windows Server 2012 R2
Rev 4.40	July 17, 2013	Updated the following sections: • Section 3.1.4, "RDMA over Converged Ethernet (RoCE)", on page 34 • Section 5, "OpenSM - Subnet Manager", on page 80 • Section 11.4.19, "NTttcp", on page 167 • Section 5, "Troubleshooting", on page 139 Added the following sections: Appendix A: "NVGRE Configuration Scripts Examples," on page 147

Table 1 - Revision History

Document Revision	Date	Changes
	June 10, 2013	Updated the following sections: • Section 2.9, "Downloading Mellanox Firmware Tools", on page 30 • Section D, "InfiniBand Fabric Utilities", on page 117 • Section 5, "Troubleshooting", on page 139 • Section 1.2, "WinOF Set of Documentation", on page 14 • Section , "Options", on page 84 Added the following sections: • "perf_tuning"Appendix ,"Synopsis," on page 84 • Section 2.10.1, "Upgrading Firmware Manually", on page 31 • Section 3.1.4.2, "RoCE Configuration", on page 36 • Section 7.4, "Adapter Proprietary Performance Counters", on page 90
Rev 4.2	October 20, 2012	Added the following sections: • Section 3.4.1, "Deploying Windows Server 2012 and Above with SMB Direct", on page 64, and its subsections • Section 3.1.6, "Header Data Split", on page 46 • Section 11.2, "part_man - Virtual IPoIB Port Creation Utility", on page 107 Updated Section 7, "Performance Tuning", on page 81
Rev 3.2.0	July 23, 2012	• No changes
Rev 3.1.0	May 21, 2012	• Added section Tuning the IPoIB Network Adapter • Added section Tuning the Ethernet Network Adapter • Added section Performance tuning tool application • Removed section Tuning the Network Adapter • Removed section part_man • Removed section ibdiagnet

Table 1 - Revision History

Document Revision	Date	Changes
Rev 3.0.0	February 08, 2012	• Added section RDMA over Converged Ethernet (RoCE) and its subsections • Added section Hyper-V with VMQ • Added section Network Driver Interface Specification (NDIS) • Added section Header Data Split • Added section Auto Sensing • Added section Adapter Teaming • Added section Port Protocol Configuration • Added section Advanced Configuration for InfiniBand Driver • Added section Advanced Configuration for Ethernet Driver • Added section Updated section Tunable Performance Parameters • Added section Merged Ethernet and InfiniBand features sections • Removed section Sockets Direct Protocol and its subsections • Removed section Winsock Direct and Protocol and its subsections • Removed section Added ConnectX®-3 support • Removed section IPoIB Drivers Overview • Removed section Booting Windows from an iSCSI Target
Rev 2.1.3	January 28. 2011	Complete restructure
Rev 2.1.2	October 10, 2010	• Removed section Debug Options. • Updated Section 3, “Uninstalling Mellanox VPI Driver,” on page 11 • Added Section 6, “InfiniBand Fabric,” on page 38 and its subsections • Added Section 6.3, “InfiniBand Fabric Performance Utilities,” on page 71 and its subsections
Rev 2.1.1.1	July 14, 2010	• Removed all references of InfiniHost® adapter since it is not supported starting with WinOF VPI v2.1.1
Rev 2.1.1	May 2010	First release

About this Manual

Scope

The document describes WinOF Rev 4.80.50000 features, performance, diagnostic, tools content and configuration. Additionally, this document provides information on various performance tools supplied with this version.

Intended Audience

This manual is intended for system administrators responsible for the installation, configuration, management and maintenance of the software and hardware of VPI (InfiniBand, Ethernet) adapter cards. It is also intended for application developers.

Documentation Conventions

Table 2 - Documentation Conventions

Description	Convention	Example
File names	file.extension	
Directory names	directory	
Commands and their parameters	command param1	mts3610-1 > show hosts
Required item	< >	
Optional item	[ ]	
Mutually exclusive parameters	{ p1, p2, p3 } or {p1 | p2 | p3}	
Optional mutually exclusive parameters	[p1 | p2 | p3]	
Variables for which users supply specific values	Italic font	*enable*
Emphasized words	Italic font	*These are emphasized words*
Note	 <text>	 This is a note..
Warning	 <text>	 May result in system instability.

Common Abbreviations and Acronyms

Table 3 - Abbreviations and Acronyms

Abbreviation / Acronym	Whole Word / Description
B	(Capital) 'B' is used to indicate size in bytes or multiples of bytes (e.g., 1KB = 1024 bytes, and 1MB = 1048576 bytes)
b	(Small) 'b' is used to indicate size in bits or multiples of bits (e.g., 1Kb = 1024 bits)
FW	Firmware
HCA	Host Channel Adapter
HW	Hardware
IB	InfiniBand
LSB	Least significant *byte*
lsb	Least significant *bit*
MSB	Most significant *byte*
msb	Most significant bit
NIC	Network Interface Card
NVGRE	Network Virtualization using Generic Routing Encapsulation
SW	Software
VPI	Virtual Protocol Interconnect
IPoIB	IP over InfiniBand
PFC	Priority Flow Control
PR	Path Record
RDS	Reliable Datagram Sockets
RoCE	RDMA over Converged Ethernet
SL	Service Level
MPI	Message Passing Interface
EoIB	Ethernet over InfiniBand
QoS	Quality of Service
ULP	Upper Level Protocol
VL	Virtual Lane

Related Documents

Table 4 - Related Documents

Document	Description
MFT User Manual	Describes the set of firmware management tools for a single InfiniBand node. MFT can be used for: • Generating a standard or customized Mellanox firmware image Querying for firmware information • Burning a firmware image to a single InfiniBand node • Enabling changing card configuration to support SRIOV
WinOF Release Notes	For possible software issues, please refer to WinOF Release Notes.
MLNX OFED User Manual	For more information on SR-IOV over KVM, please refer to OFED User Manual.

1 Introduction

This User Manual describes installation, configuration and operation of Mellanox WinOF driver Rev 4.80.50000 package.

Mellanox WinOF is composed of several software modules that contain InfiniBand and Ethernet drivers. The Mellanox WinOF driver supports 10, 40 or 56 Gb/s Ethernet, and 40 or 56 Gb/s InfiniBand network ports. The port type is determined upon boot based on card capabilities and user settings.

The Mellanox VPI WinOF driver release introduces the following capabilities:

- Support for Single and Dual port Adapters
- Up to 16 Rx queues per port
- Rx steering mode (RSS)
- Hardware Tx/Rx checksum calculation
- Large Send off-load (i.e., TCP Segmentation Off-load)
- Hardware multicast filtering
- Adaptive interrupt moderation
- Support for MSI-X interrupts
- Support for Auto-Sensing of Link level protocol
- NDK with SMB-Direct
- NDv1 and v2 API support in user space
- VMQ for Hypervisor
- CIM and PowerShell

Ethernet Only:

- Hardware VLAN filtering
- Header Data Split
- RDMA over Converged Ethernet (RoCE MAC Based)
- RoCE IP Based
- RoCEv2 in ConnectX®-3 Pro
- DSCP over IPv4
- NVGRE hardware off-load in ConnectX®-3 Pro
- Ports TX arbitration/Bandwidth allocation per port
- Enhanced Transmission Selection (ETS)
- SRIOV Ethernet on Windows 2012R2 Hypervisor with Windows 2012 and above guests.

InfiniBand Only:

- SRIOV over KVM Hypervisor
- Diagnostic tools

For the complete list of Ethernet and InfiniBand Known Issues and Limitations, WinOF Release Notes (www.mellanox.com -> Products -> Software -> InfiniBand/VPI Drivers -> Windows SW/ Drivers).

1.1 Supplied Packages

Mellanox WinOF driver Rev 4.80.50000 includes the following package:

- MLNX_VPI_WinOF-<version>_All_<OS>_<arch>.exe:

  In this package, the port default is auto, RoCE is enabled

1.2 WinOF Set of Documentation

Under <installation_directory>\Documentation:

- License file
- User Manual (this document)
- MLNX_VPI_WinOF Release Notes

1.3 Windows MPI (MS-MPI)

Message Passing Interface (MPI) is meant to provide virtual topology, synchronization, and communication functionality between a set of processes. MPI enables running one process on several hosts.

- Windows MPI runs over the following protocols:
  - Sockets (Ethernet)
  - Network Direct (ND)

For further details on MPI, please refer to Appendix B,“Windows MPI (MS-MPI),” on page 150.

2 Installation

2.1 Hardware and Software Requirements

Table 5 - Hardware and Software Requirements

Description[a]	Package
Windows Server 2008 R2 (64 bit only)	MLNX_VPI_WinOF-4.80_All_win2008R2_x64.exe
Windows Server 2012 (64 bit only)	MLNX_VPI_WinOF-4.80_All_win2012_x64.exe
Windows Server 2012 R2 (64 bit only)	MLNX_VPI_WinOF-4.80_All_win2012R2_x64.exe
Windows 8.1 Client (64 bit only)	MLNX_VPI_WinOF-4.80_All_win8.1_x64.exe
Windows 7 Client (64 bit only)	MLNX_VPI_WinOF-4.80_All_win7_x64.exe

a. The Operating System listed above must run with administrator privileges.

Required Disk Space for Installation is 100MB

2.2 Downloading Mellanox WinOF Driver

Follow these steps to download the .exe according to your Operating System.

Step 1. Verify the machine architecture.

For Windows Server 2008 R2

1. Open a CMD console (Click start-->Run and enter CMD).
2. Enter the following command.

```
> echo %PROCESSOR_ARCHITECTURE%
```

On an x64 (64-bit) machine, the output will be “AMD64”.

For Windows Server 2012 / 2012 R2

1. To go to the Start menu.

   Position your mouse in the bottom-right corner of the Remote Desktop of your screen.
2. Open a CMD console (Click Task Manager-->File --> Run new task --> and enter CMD).
3. Enter the following command.

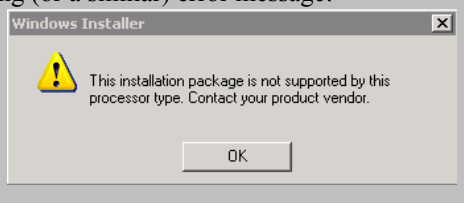
```
> echo %PROCESSOR_ARCHITECTURE%
```

On an x64 (64-bit) machine, the output will be “AMD64”.

Step 2. Go to the Mellanox WinOF web page at http://www.mellanox.com > Products > InfiniBand/VPI Drivers => Windows SW/Drivers.

Step 3. Download the .exe image according to the architecture of your machine (see Step 1) and the operating system. The name of the .exe is in the following format MLNX_VPI_WinOF-<version>_All_<OS>_<arch>.exe.

Installing the incorrect .exe file is prohibited. If you do so, an error message will be displayed. For example, if you try to install a 64-bit .exe on a 32-bit machine, the wizard will display the following (or a similar) error message:



2.3 Extracting Files Without Running Installation

To extract the files without running installation, perform the following steps.

Step 1. Open a CMD console **[Windows Server 2008 R2] -** Click Start-->Run and enter CMD.

[Windows Server 2012 / 2012 R2] - Click Start --> Task Manager-->File --> Run new task --> and enter CMD.

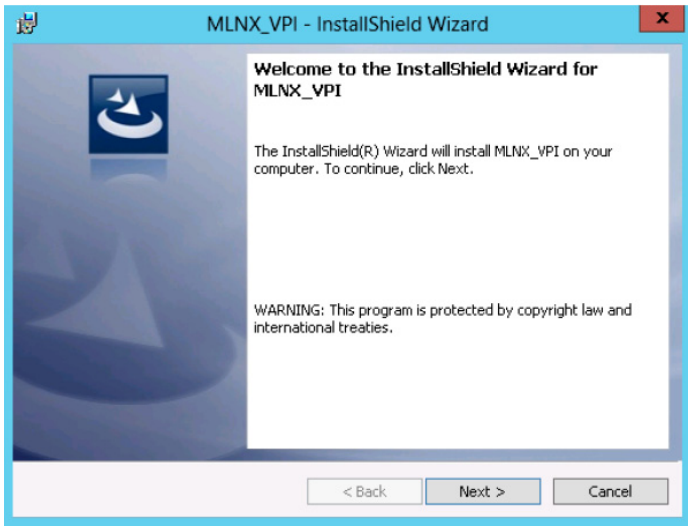
Step 2. Extract the driver and the tools:

```
> MLNX_VPI_WinOF-<version>_All_<OS>_<arch>.exe /a
```

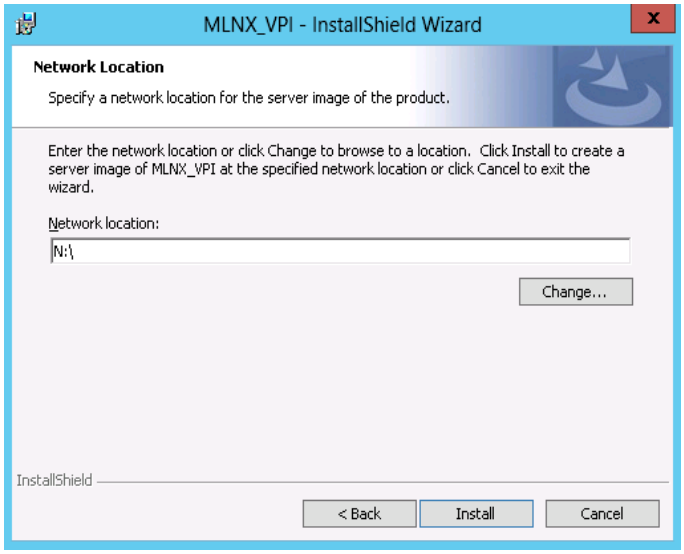
- To extract only the driver files.

```
> MLNX_VPI_WinOF-<version>_All_<OS>_<arch>.exe /a /vMT_DRIVERS_ONLY=1
```

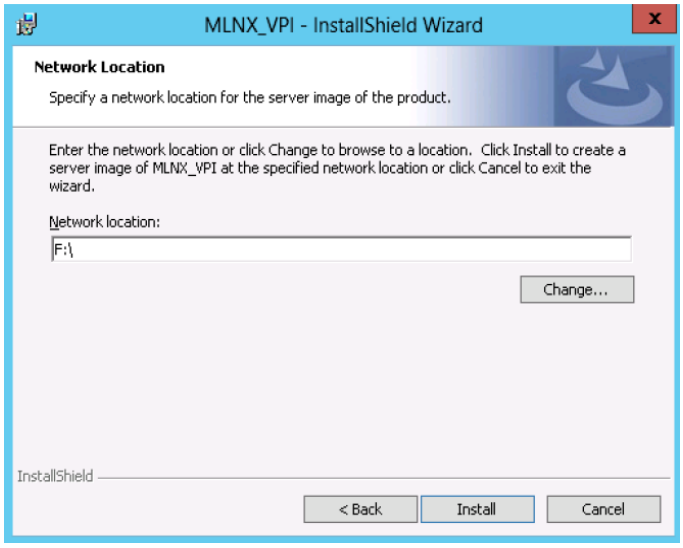
Step 3. Click Next to create a server image.



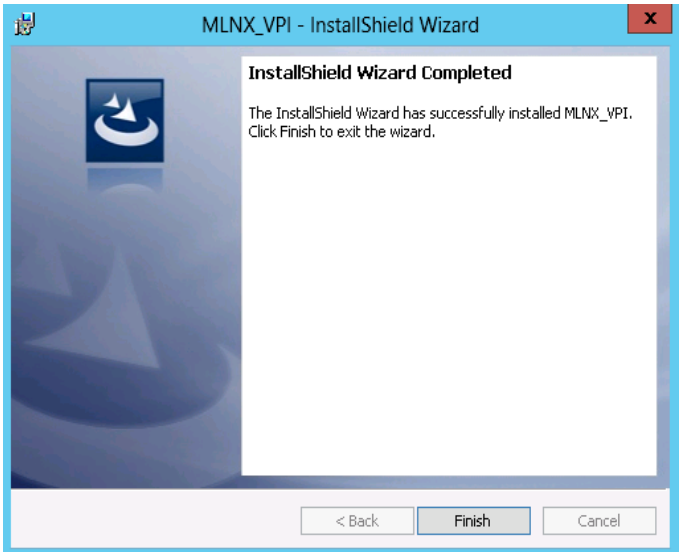
Step 4. Click Change and specify the location in which the files are extracted to.



Step 5. Click Install to extract this folder, or click Change to install to a different folder.



Step 6. To complete the extraction, click Finish.



2.4 Installing Mellanox WinOF Driver

This section provides instructions for two types of installation procedures:

- "Attended Installation"

  An installation procedure that requires frequent user intervention.
- "Unattended Installation"

  An automated installation procedure that requires no user intervention.

Both Attended and Unattended installations require administrator privileges.

2.4.1 Attended Installation

The following is an example of a MLNX_WinOF_win2012 x64 installation session.

Step 1. Double click the .exe and follow the GUI instructions to install MLNX_WinOF.

As of MLNX WinOF v4.55, the log option is enabled automatically.
The default path of the log is: `%LOCALAPPDATA%\MLNX_WinOF.log0`

Step 2. [Optional] Manually configure your setup to contain the logs option.

```
> MLNX_VPI_WinOF-4.80_All_win2012_x64.exe /v"/l*vx [LogFile]"
```

Step 3. [Optional] If you do not want to upgrade your firmware version[1].

```
> MLNX_VPI_WinOF-4.80_All_win2012_x64.exe /v" MT_SKIPFWUPGRD=1"
```

1. MT_SKIPFWUPGRD default value is False

Step 4. [Optional] If you want to control the installation of the WMI/CIM provider[1].

```
> MLNX_VPI_WinOF-4.80_All_win2012_x64.exe /v" MT_WMI=1"
```

Step 5. [Optional] If you want to control whether to restore network configuration or not[2].

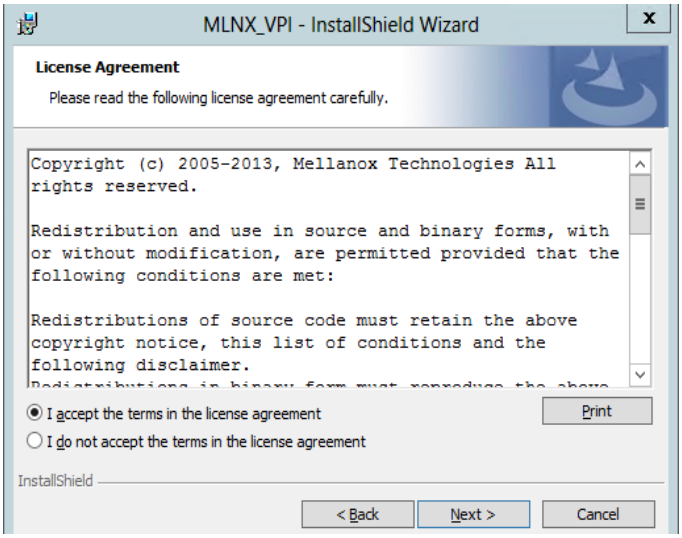
```
> MLNX_VPI_WinOF-4.80_All_win2012_x64.exe /v" MT_RESTORECONF=1"
```

For further help, please run:

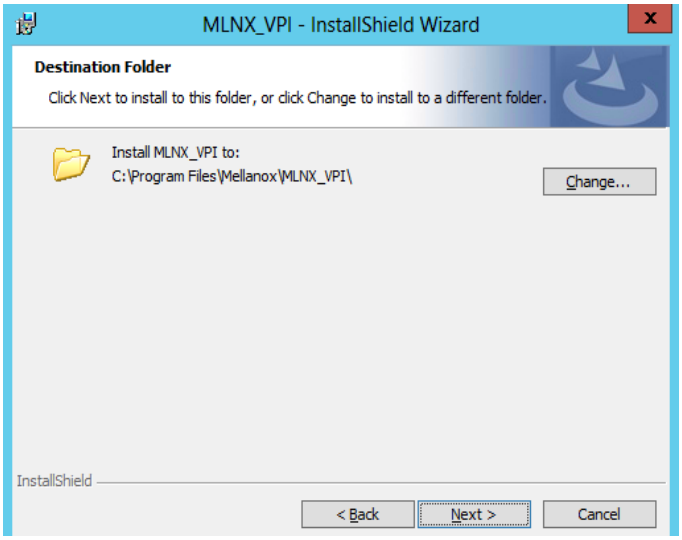
```
> MLNX_VPI_WinOF-4.80_All_win2012_x64.exe /v" /h"
```

Step 6. Click Next in the Welcome screen.

Step 7. Read then accept the license agreement and click Next.



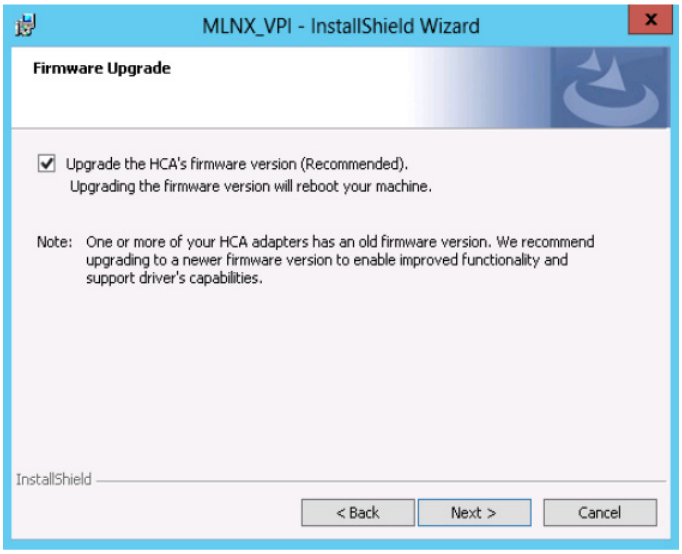
Step 8. Select the target folder for the installation.



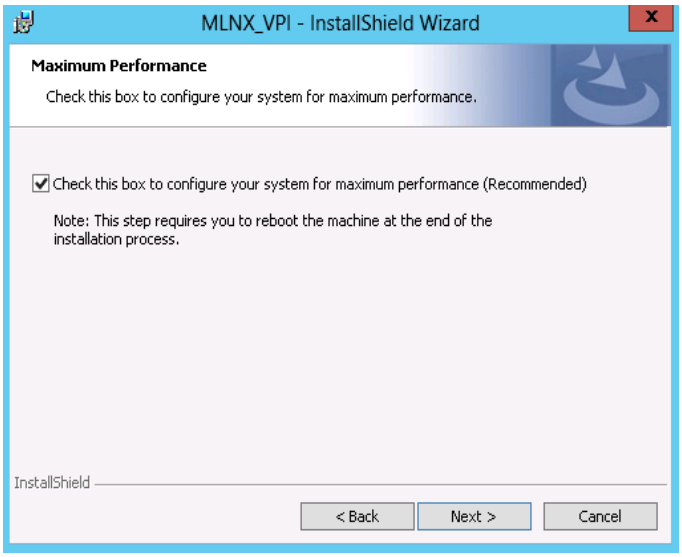
1. MT_WMI default value is True
2. MT_RESTORECONF default value is True

Step 9. The firmware upgrade screen will be displayed in the following cases:

- If the user has an OEM card, in this case the firmware will not be updated.
- If the user has a standard Mellanox card with an older firmware version, the firmware will be updated accordingly. However, if the user has both OEM card and Mellanox card, only Mellanox card will be updated.

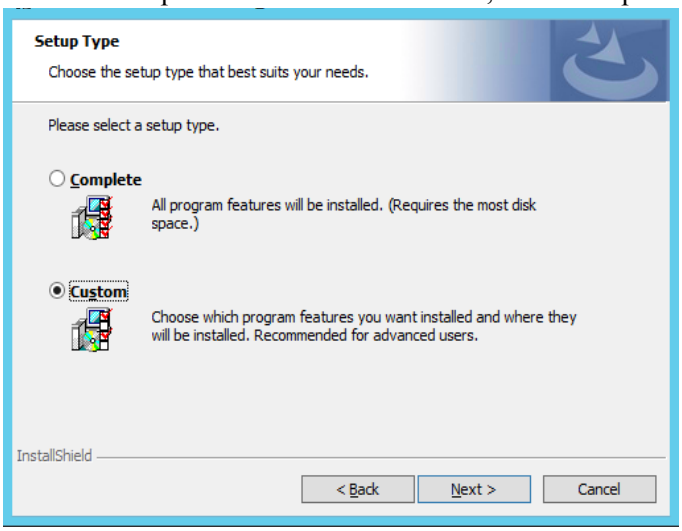


Step 10. Configure your system for maximum performance by checking the maximum performance box.



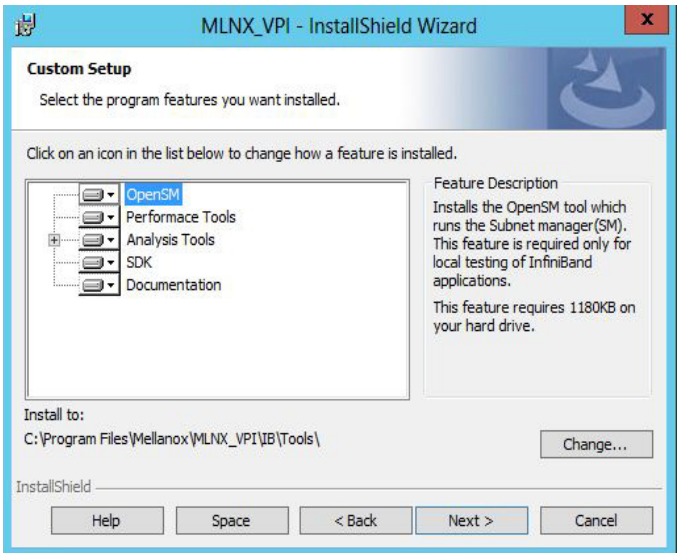
This step requires rebooting your machine at the end of the installation.

Step 11. Select a Complete or Custom installation, follow Step a and on, on page 21.

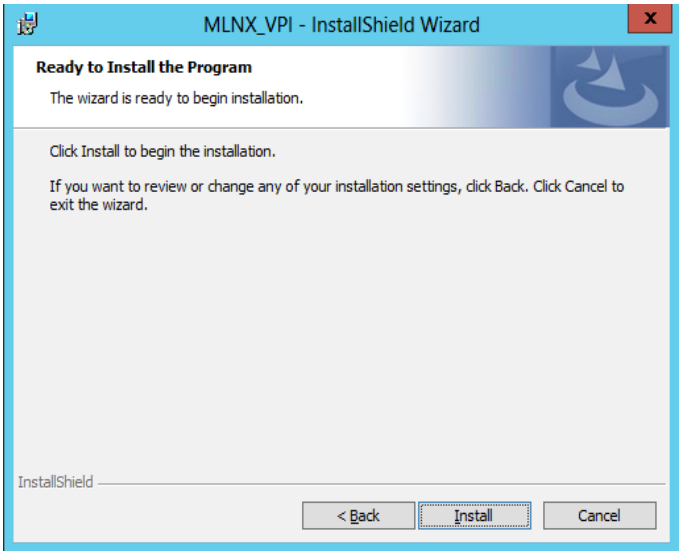


a.Select the desired feature to install:

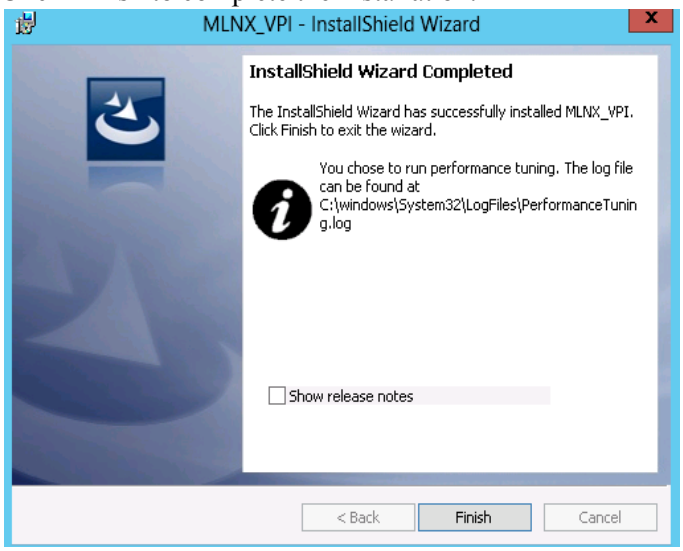
- OpenSM - installs Windows OpenSM that is required to manage the subnet from a host. OpenSM is part of the driver and installed automatically.
- Performances tools - install the performance tools that are used to measure the InfiniBand performance in user environment.
- Analyze tools - install the tools that can be used either to diagnosed or analyzed the InfiniBand environment.
- SDK - contains the libraries and DLLs for developing InfiniBand application over IBAL.
- Documentation - contains the User Manual and Installation Guide.



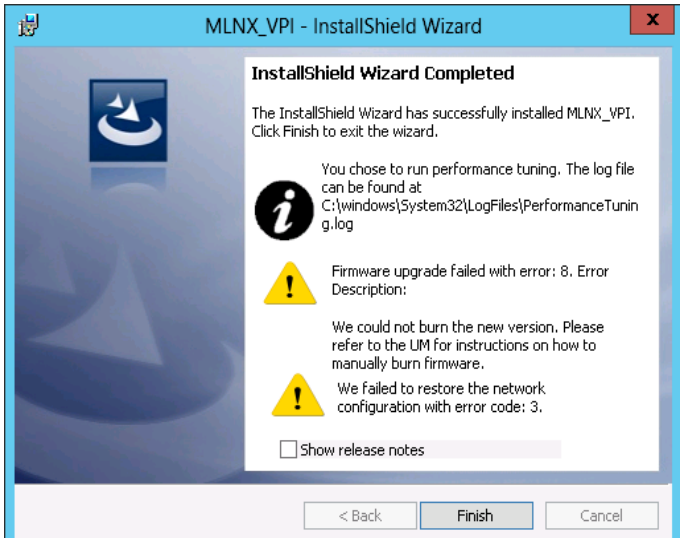
b. Click Install to start the installation.



Step 12. Click Finish to complete the installation.



- If the firmware upgrade and the restore of the network configuration failed, the following message will be displayed.



2.4.2 Unattended Installation

The following is an example of a MLNX_WinOF_win2012 x64 unattended installation session.

Step 1. Open a CMD console

[Windows Server 2008 R2] - Click Start-->Run and enter CMD.

[Windows Server 2012 / 2012 R2] - Click Start --> Task Manager-->File --> Run new task --> and enter CMD.

Step 2. Install the driver. Run:

```
> MLNX_VPI_WinOF-4.80_All_win2012_x64.exe /S /v"/qn"
```

Step 3. [Optional] Manually configure your setup to contain the logs option:

```
> MLNX_VPI_WinOF-4.80_All_win2012_x64.exe /S /v"/qn" /v"/l*vx [LogFile]"
```

Starting from MLNX WinOF v4.55, the log option is enabled automatically. The default path of the log is: `%LOCALAPPDATA%\MLNX_WinOF.log0`

Step 4. [Optional] If you do not want to upgrade your firmware version[1].

```
> MLNX_VPI_WinOF-4.80_All_win2012_x64.exe /v" MT_SKIPFWUPGRD=1"
```

Step 5. [Optional] If you want to control the installation of the WMI/CIM provider[2].

```
> MLNX_VPI_WinOF-4.80_All_win2012_x64.exe /v" /MT_WMI=1"
```

Step 6. [Optional] If you want to control whether to restore network configuration or not[3].

```
> MLNX_VPI_WinOF-4.80_All_win2012_x64.exe /v" MT_RESTORECONF=1"
```

1. MT_SKIPFWUPGRD default value is False
2. MT_WMI default value is True
3. MT_RESTORECONF default value is True

For further help, please run:

```
> MLNX_VPI_WinOF-4.80_All_win2012_x64.exe /v" /h"
```

Step 7. [Optional] if you want to control whether to execute performance tuning or not[1].

```
> MLNX_VPI_WinOF_4.80_All_win2012_x64.exe /vPERFCHECK=0 /vPERFCHECK=0
```

Step 8. [Optional] if you want to control whether to install ND provider or not[2].

```
> MLNX_VPI_WinOF_4.80_All_win2012_x64.exe /vMT_NDPROPERTY=1
```

2.5 Installation Results

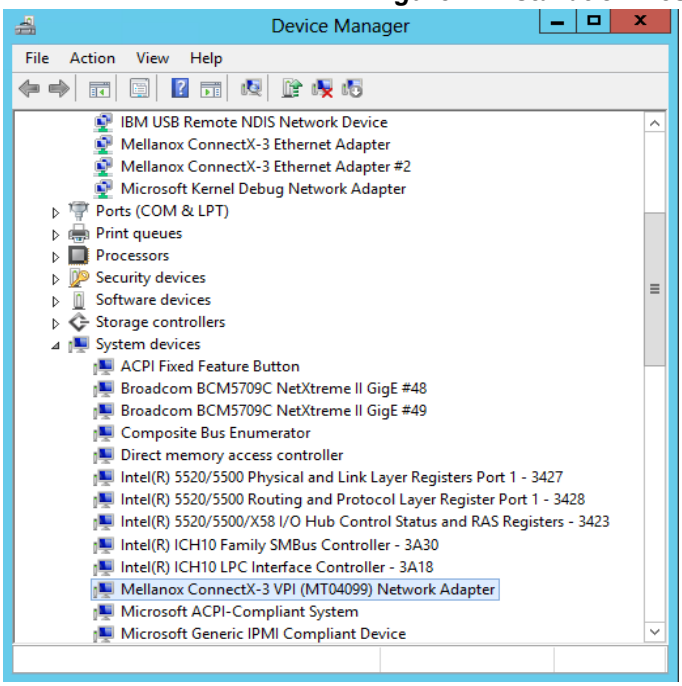
Upon installation completion, you can verify the successful addition of the network card(s) through the Device Manager.

Upon installation completion, the inf files can be located at:

- %ProgramFiles%\Mellanox\MLNX_VPI\ETH
- %ProgramFiles%\Mellanox\MLNX_VPI\HW\mlx4_bus
- %ProgramFiles%\Mellanox\MLNX_VPI\IB\IPoIB

To see the Mellanox network adapter device, and the Ethernet or IPoIB network device (depending on the used card) for each port, display the Device Manager and expand “System devices” or “Network adapters”.

Figure 1: Installation Results



1. PERFCHECK default value is True
2. MT_NDPROPERTY default value is True

2.6 Uninstalling Mellanox WinOF Driver

2.6.1 Attended Uninstallation

➢ ***To uninstall MLNX_WinOF on a single node:***

1. Click Start-> Control Panel-> Programs and Features-> MLNX_VPI-> Uninstall.
   (NOTE: This requires elevated administrator privileges – see Section 1.1, "Supplied Packages", on page 14 for details.)
2. Double click the .exe and follow the instructions of the install wizard.
3. Click Start -> All Programs -> Mellanox Technologies -> MLNX_WinOF -> Uninstall MLNX_WinOF.

2.6.2 Unattended Uninstallation

➢ ***To uninstall MLNX_WinOF in unattended mode:***

Step 1. Open a CMD console

[Windows Server 2008 R2] - Click Start-->Run and enter CMD.

[Windows Server 2012 / 2012 R2] - Click Start --> Task Manager-->File --> Run new task --> and enter CMD.

Step 2. Uninstall the driver. Run:

```
> MLNX_VPI_WinOF-4.80_All_win2012_x64.exe /S /x /v"/qn"
```

2.7 Firmware Upgrade

If the machine has a standard Mellanox card with an older firmware version, the firmware will be updated automatically as part of the installation of the WinOF package.

For information on how to upgrade firmware manually please refer to MFT User Manual:
www.mellanox.com ->Products -> Adapter IB/VPI SW ->Firmware Tools

2.8 Upgrading Mellanox WinOF Driver

The upgrade process differs between various Operating Systems.

- Windows Server 2008 R2:

  When upgrading from WinOF version 3.2.0 to version 4.40 and above, the MLNX_WinOF driver upgrades the driver automatically by uninstalling the previous version and installing the new driver. The existing configuration files are not saved upon driver upgrade.
- Windows Server 2012 and above:
  - When upgrading from WinOF version 4.2 to version 4.40 and above, the MLNX_WinOF driver does not completely uninstall the previous version, but rather upgrades only the components that require upgrade. The network configuration is saved upon driver upgrade.
  - When upgrading from Inbox or any other version, the network configuration is automatically saved upon driver upgrade.

2.9 Booting Windows from an iSCSI Target

2.9.1 Configuring the WDS, DHCP and iSCSI Servers

2.9.1.1 Configuring the WDS Server

➢ ***To configure the WDS server:***

1. Install the WDS server.
2. Extract the Mellanox drivers to a local directory using the '-a' parameter.

   For boot over Ethernet, when using adapter cards with older firmware version than 2.30.8000, you need to extract the PXE package, otherwise use Mellanox WinOF VPI package.

   Example:

   ```
   Mellanox.msi.exe -a
   ```

3. Add the Mellanox driver to boot.wim[1].

   ```
   dism /Mount-Wim /WimFile:boot.wim /index:2 /MountDir:mnt
   dism /Image:mnt /Add-Driver /Driver:drivers /recurse
   dism /Unmount-Wim /MountDir:mnt /commit
   ```

4. Add the Mellanox driver to install.wim[2].

   ```
   dism /Mount-Wim /WimFile:install.wim /index:4 /MountDir:mnt
   dism /Image:mnt /Add-Driver /Driver:drivers /recurse
   dism /Unmount-Wim /MountDir:mnt /commit
   ```

5. Add the new boot and install images to WDS.

For additional details on WDS, please refer to:

http://technet.microsoft.com/en-us/library/jj648426.aspx

2.9.1.2 Configuring iSCSI Target

➢ ***To configure iSCSI Target:***

1. Install iSCSI Target (e.g StartWind).
2. Add to the iSCSI target initiators the IP addresses of the iSCSI clients.

2.9.1.3 Configuring the DHCP Server

➢ ***To configure the DHCP server:***

1. Install a DHCP server.
2. Add to IPv4 a new scope.
3. Add iSCSI boot client identifier (MAC/GUID) to the DHCP reservation.

---

1. Use `'index:2'` for Windows setup and `'index:1'` for WinPE.
2. When adding the Mellanox driver to install.wim, verify you are using the appropriate index for your OS flavor. To check the OS run `'imagex /info install.win'`.

4. Add to the reserved IP address the following options:

Table 6 - Reserved IP Address Options

Option	Name	Value
017	Root Path	**iscsi:11.4.12.65::::iqn:2011-01:iscsiboot** Assuming the iSCSI target IP is: **11.4.12.65** and the Target Name: **iqn:2011-01:iscsiboot**
060	PXEClient	PXEClient
066	Boot Server Host Name	WDS server IP address
067	Boot File Name	boot\x86\wdsnbp.com

2.9.2 Configuring the Client Machine

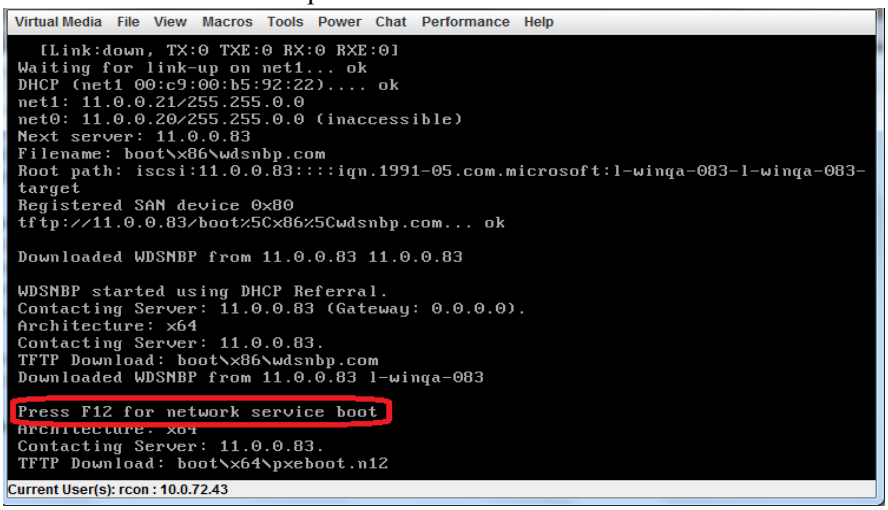
➢ ***To configuring your client:***

1. Verify the Mellanox adapter card is burned with the correct Mellanox FlexBoot version.

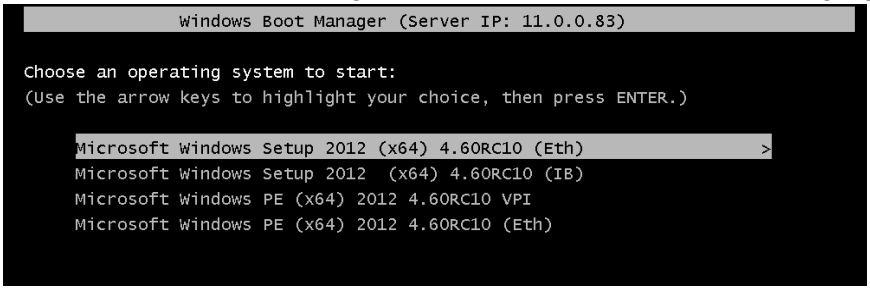
   For boot over Ethernet, when using adapter cards with older firmware version than 2.30.8000, you need to burn the adapter card with Ethernet FlexBoot, otherwise use the VPI FlexBoot.
2. Verify the Mellanox adapter card is burned with the correct firmware version.
3. Set the “Mellanox Adapter Card” as the first boot device in the BIOS settings boot order.

2.9.3 Installing iSCSI

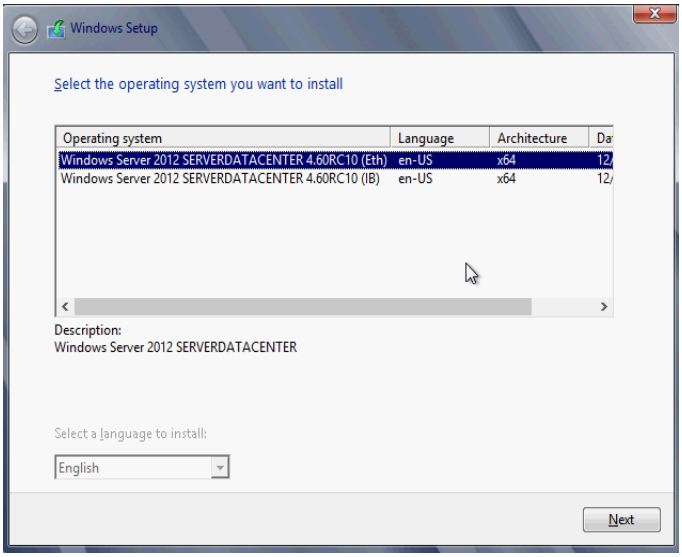
1. Reboot your iSCSI client.
2. Press F12 when asked to proceed to iSCSI boot.



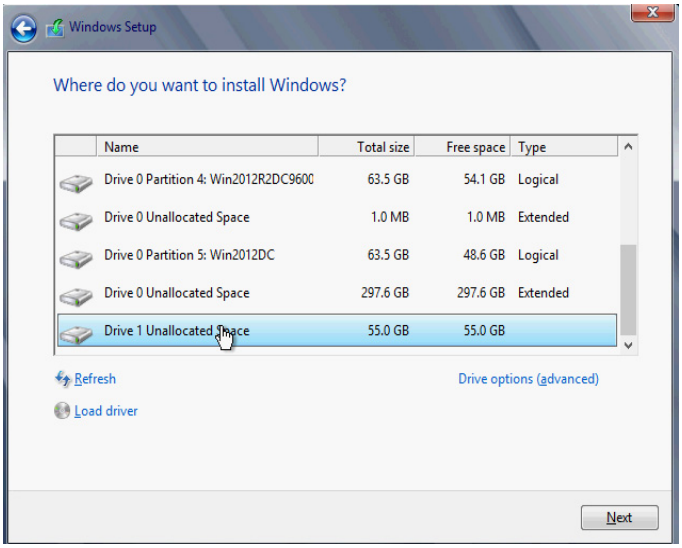
3. Choose the relevant boot image from the list of all available boot images presented.



4. Choose the Operating System you wish to install.



5. Run the Windows Setup Wizard.
6. Choose iSCSI target drive to install Windows and follow the instructions presented by the installation Wizard.



Installation process will start once completing all the required steps in the Wizard, the Client will reboot and will boot from the iSCSI target.

3 Features Overview and Configuration

Once you have installed Mellanox WinOF VPI package, you can perform various modifications to your driver to make it suitable for your system's needs

Changes made to the Windows registry happen immediately, and no backup is automatically made.

Do ***not*** edit the Windows registry unless you are confident regarding the changes.

3.1 Ethernet Network

3.1.1 Port Configuration

3.1.1.1 Auto Sensing

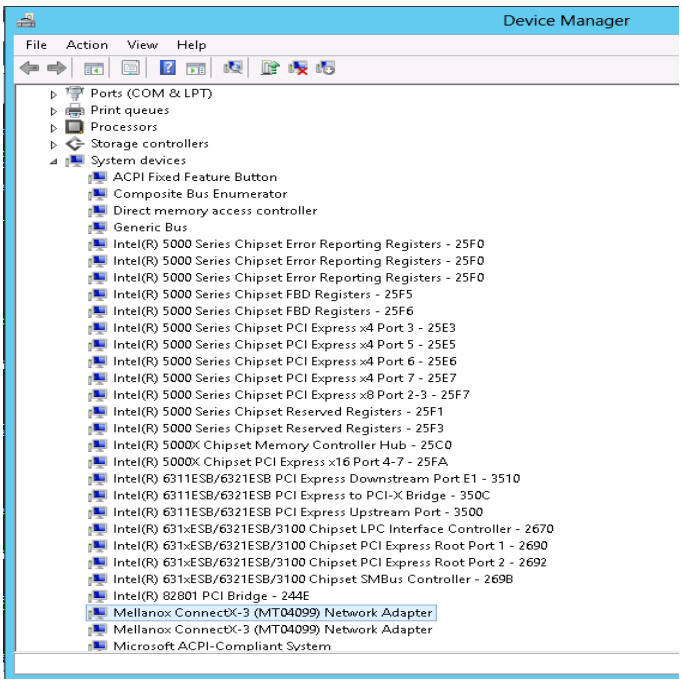
Auto Sensing enables the NIC to automatically sense the link type (InfiniBand or Ethernet) based on the cable connected to the port and load the appropriate driver stack (InfiniBand or Ethernet).

Auto Sensing is performed only when rebooting the machine or after disabling/enabling the mlx4_bus interface from the Device Manager. Hence, if you replace cables during the runtime, the NIC will not perform Auto Sensing.

For further information on how to configure it, please refer to Section 3.1.1.2, "Port Protocol Configuration", on page 29.

3.1.1.2 Port Protocol Configuration

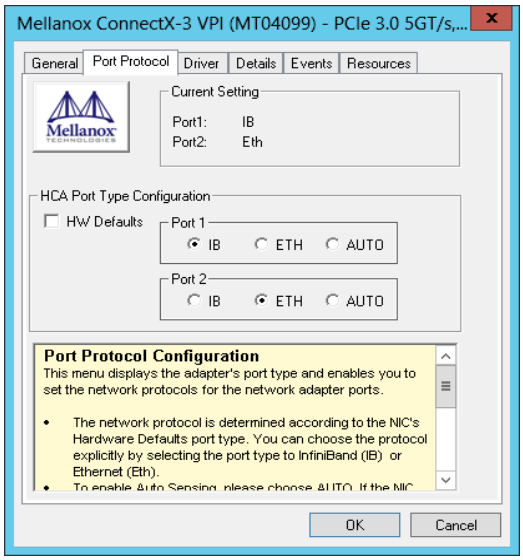
Step 1. Display the Device Manager and expand "System devices".



Step 2. Right-click on the Mellanox ConnectX Ethernet network adapter and left-click Properties. Select the Port Protocol tab from the Properties window.

The "Port Protocol" tab is displayed only if the NIC is a VPI (IB and ETH).

The figure below is an example of the displayed Port Protocol window for a dual port VPI adapter card.



Step 3. In this step, you can perform the following functions:

- If you choose the HW Defaults option, the port protocols will be determined according to the NIC's hardware default values.
- Choose the desired port protocol for the available port(s). If you choose IB or ETH, both ends of the connection must be of the same type (IB or ETH).
- Enable Auto Sensing by checking the AUTO checkbox. If the NIC does not support Auto Sensing, the AUTO option will be grayed out.

If you choose AUTO, the current setting will indicate the actual port settings: IB or ETH.

For firmware 2.32.5000 and above, there is an option to set port personality using mlxfwconfig tool. For further details please refer to MFT User Manual

3.1.2 Assigning Port IP After Installation

By default, your machine is configured to obtain an automatic IP address via a DHCP server. In some cases, the DHCP server may require the MAC address of the network adapter installed in your machine.

➢ ***To obtain the MAC address:***

Step 1. Open a CMD console

[Windows Server 2008 R2] - Click Start-->Run and enter CMD.

[Windows Server 2012 / 2012 R2] - Click Start --> Task Manager-->File --> Run new task --> and enter CMD.

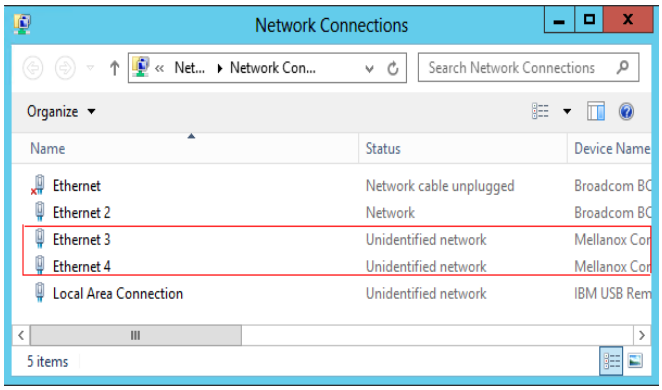
Step 2. Display the MAC address as “Physical Address”

```
> ipconfig /all
```

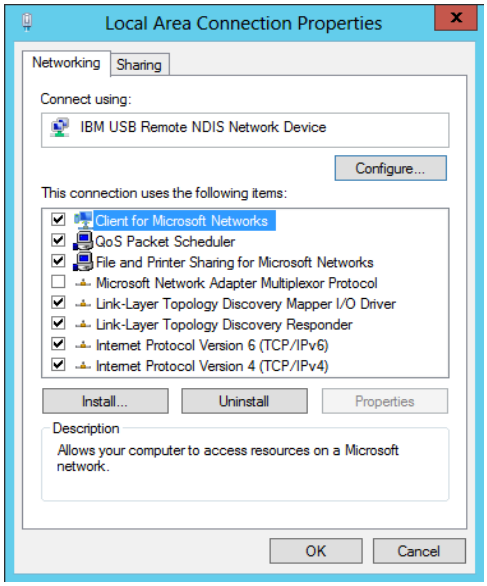
Configuring a static IP is the same for both IPoIB and Ethernet adapters.

➢ ***To assign a static IP address to a network port after installation:***

Step 1. Open the Network Connections window. Locate Local Area Connections with Mellanox devices.

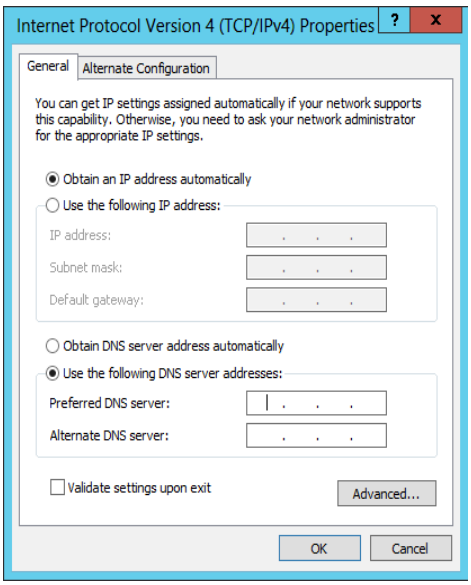


Step 2. Right-click a Mellanox Local Area Connection and left-click Properties.



Step 3. Select Internet Protocol Version 4 (TCP/IPv4) from the scroll list and click Properties.

Step 4. Select the “Use the following IP address:” radio button and enter the desired IP information.



Step 5. Click OK.

Step 6. Close the Local Area Connection dialog.

Step 7. Verify the IP configuration by running ‘ipconfig’ from a CMD console.

```
> ipconfig
...
Ethernet adapter Local Area Connection 4:

   Connection-specific DNS Suffix  . :
   IP Address. . . . . . . . . . . . : 11.4.12.63
   Subnet Mask . . . . . . . . . . . : 255.255.0.0
   Default Gateway . . . . . . . . . :
...
```

3.1.3 56GbE Link Speed

Mellanox offers proprietary speed of 56GbE link speed over FDR systems. To achieve this, only the switch, supporting this speed, must be configured to enable it. The NIC on the other hand, auto-detects this configuration automatically.

➢ ***To achieve 56GbE link speed over SwitchX® Based Switch System***

Make sure your switch supports 56GbE and that you have the relevant switch license installed.

Step 1. Set the system profile to be `eth-single-switch`, and reset the system:

```
switch (config) # system profile eth-single-profile
```

Step 2. Set the speed for the desired interface to 56GbE as follows. For example (for interface 1/1):

```
switch (config) # interface ethernet 1/1
switch (config interface ethernet 1/1) # speed 56000
switch (config interface ethernet 1/1) #
```

Step 3. Verify the speed is 56GbE.

```
switch (config) # show interface ethernet 1/1
Eth1/1
Admin state: Enabled
Operational state: Down
Description: N\A
Mac address: 00:02:c9:5d:e0:26
MTU: 1522 bytes
Flow-control: receive off send off
Actual speed: 56 Gbps
Switchport mode: access
Rx
0 frames
0 unicast frames
0 multicast frames
0 broadcast frames
0 octets
0 error frames
0 discard frames
Tx
0 frames
0 unicast frames
0 multicast frames
0 broadcast frames
0 octets
0 discard frames
switch (config) #
```

3.1.4 RDMA over Converged Ethernet (RoCE)

Remote Direct Memory Access (RDMA) is the remote memory management capability that allows server to server data movement directly between application memory without any CPU involvement. RDMA over Converged Ethernet (RoCE) is a mechanism to provide this efficient data transfer with very low latencies on loss-less Ethernet networks. With advances in data center convergence over reliable Ethernet, ConnectX® EN with RoCE uses the proven and efficient RDMA transport to provide the platform for deploying RDMA technology in mainstream data center application at 10GigE, 40GigE and 56GigE link-speed. ConnectX® EN with its hardware offload support takes advantage of this efficient RDMA transport (InfiniBand) services over Ethernet to deliver ultra-low latency for performance-critical and transaction intensive applications such as financial, database, storage, and content delivery networks. RoCE encapsulates IB transport and GRH headers in Ethernet packets bearing a dedicated ether type. While the use of GRH is optional within InfiniBand subnets, it is mandatory when using RoCE. Applications written over IB verbs should work seamlessly, but they require provisioning of GRH information when creating address vectors. The library and driver are modified to provide mapping from GID to MAC addresses required by the hardware.

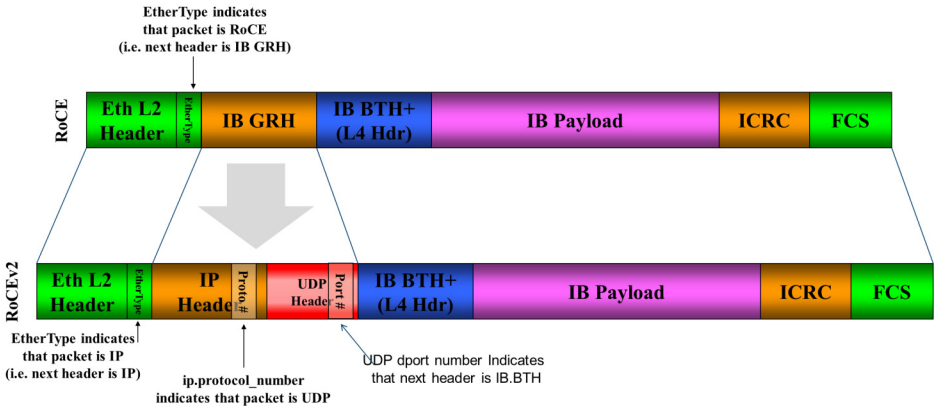
3.1.4.1 IP Routable (RoCEv2)

RoCE has two addressing modes: MAC based GIDs, and IP address based GIDs. In RoCE IP based, if the IP address changes while the system is running, the GID for the port will automatically be updated with the new IP address, using either IPv4 or IPv6.

RoCE IP based allows RoCE traffic between Windows and Linux systems, which use IP based GIDs by default.

A straightforward extension of the RoCE protocol enables traffic to operate in layer 3 environments. This capability is obtained via a simple modification of the RoCE packet format. Instead of the GRH used in RoCE, routable RoCE packets carry an IP header which allows traversal of IP L3 Routers and a UDP header that serves as a stateless encapsulation layer for the RDMA Transport Protocol Packets over IP.

Figure 2: RoCE and RoCE v2 Frame Format Differences

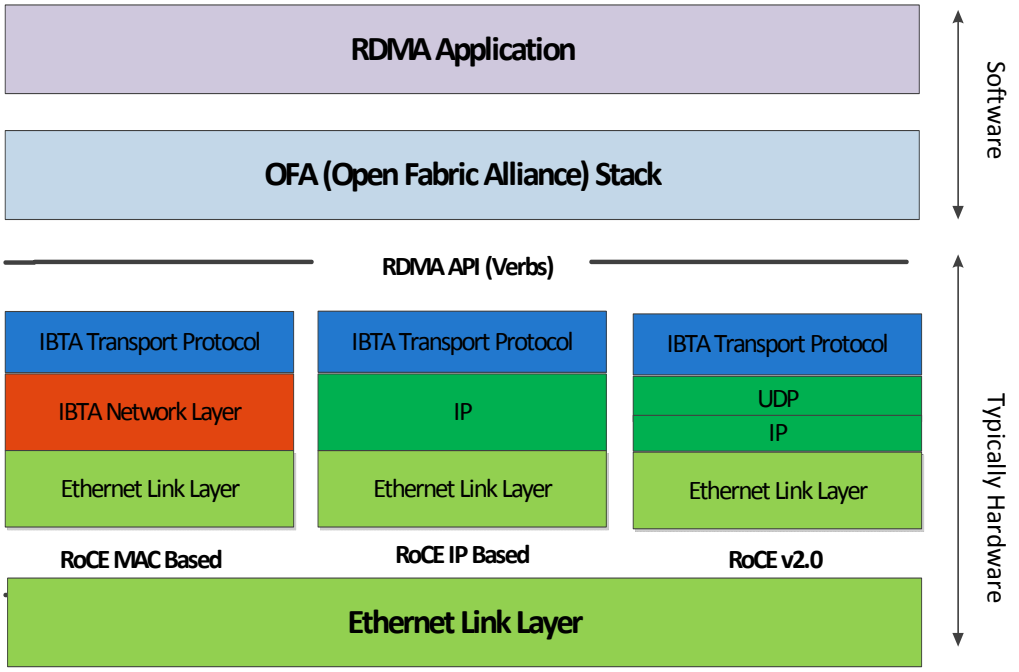


The proposed RoCEv2 packets use a well-known UDP destination port value that unequivocally distinguishes the datagram. Similar to other protocols that use UDP encapsulation, the UDP source port field is used to carry an opaque flow-identifier that allows network devices to implement packet forwarding optimizations (e.g. ECMP) while staying agnostic to the specifics of the protocol header format.

Furthermore, since this change exclusively affects the packet format on the wire, and due to the fact that with RDMA semantics packets are generated and consumed below the AP applications can seamlessly operate over any form of RDMA service (including the routable version of RoCE as shown in Figure 2,“RoCE and RoCE v2 Frame Format Differences”), in a completely transparent way[1].

1. Standard RDMA APIs are IP based already for all existing RDMA technologies

Figure 3: RoCE and RoCEv2 Protocol Stack



The fabric must use the same protocol stack in order for nodes to communicate.

The default RoCE mode in Windows is MAC based.

The default RoCE mode in Linux is IP based.

In order to communicate between Windows and Linux over RoCE, please change the RoCE mode in Windows to IP based.

3.1.4.2 RoCE Configuration

In order to function reliably, RoCE requires a form of flow control. While it is possible to use global flow control, this is normally undesirable, for performance reasons.

The normal and optimal way to use RoCE is to use Priority Flow Control (PFC). To use PFC, it must be enabled on all endpoints and switches in the flow path.

In the following section we present instructions to configure PFC on Mellanox ConnectX™ cards. There are multiple configuration steps required, all of which may be performed via PowerShell. Therefore, although we present each step individually, you may ultimately choose to write a PowerShell script to do them all in one step. Note that administrator privileges are required for these steps.

For further information, please refer to:

http://blogs.technet.com/b/josebda/archive/2012/07/31/deploying-windows-server-2012-with-smb-direct-smb-over-rdma-and-the-mellanox-connectx-3-using-10gbe-40gbe-roce-step-by-step.aspx

3.1.4.2.1 Prerequisites

The following are the driver's prerequisites in order to set or configure RoCE:

- ConnectX®-3 and ConnectX®-3 Pro firmware version 2.30.3000 or higher
- All InfiniBand verbs applications which run over InfiniBand verbs should work on RoCE links if they use GRH headers.
- Set HCA to use Ethernet protocol:
  Display the Device Manager and expand "System Devices". Please refer to Section 3.1.1.2, "Port Protocol Configuration", on page 29.

3.1.4.2.2 Configuring Windows Host

Since PFC is responsible for flow controlling at the granularity of traffic priority, it is necessary to assign different priorities to different types of network traffic.

As per RoCE configuration, all ND/NDK traffic is assigned to one or more chosen priorities, where PFC is enabled on those priorities.

Configuring Windows host requires configuring QoS. To configure QoS, please follow the procedure described in Section 3.1.8, "Configuring Quality of Service (QoS)", on page 47

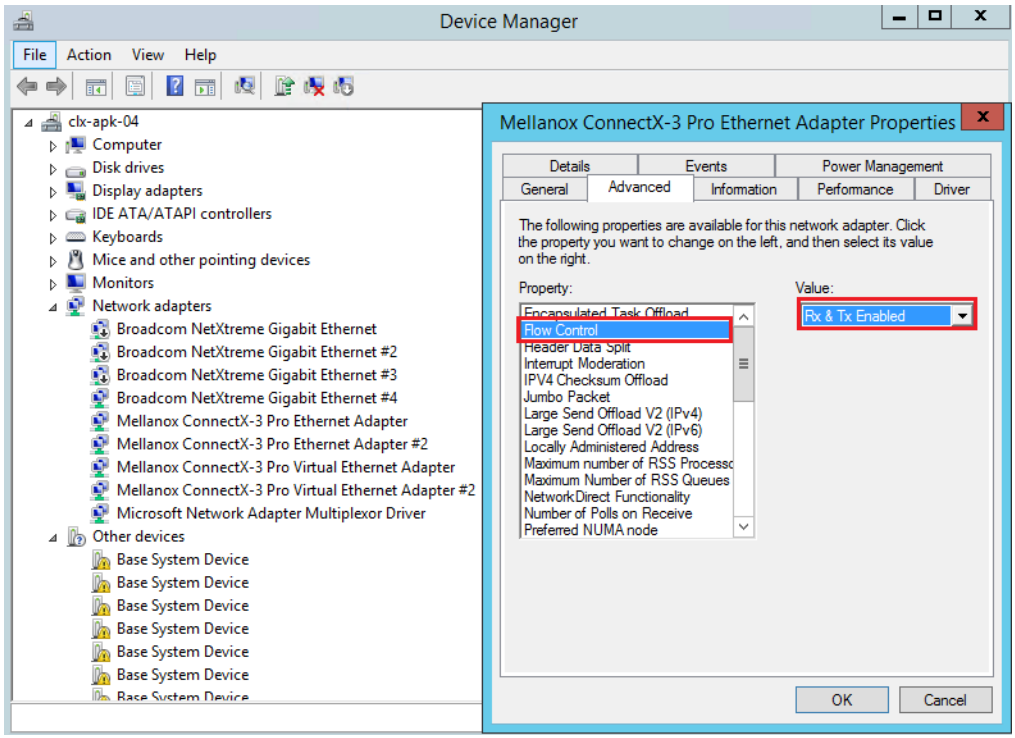
3.1.4.2.2.1 Global Pause (Flow Control)

➢ ***To use Global Pause (Flow Control) mode, disable QoS and Priority:***

```
PS $ Disable-NetQosFlowControl
PS $ Disable-NetAdapterQos <interface name>
```

➢ ***To confirm flow control is enabled in adapter parameters:***

Device manager-> Network adapters-> Mellanox ConnectX-3 Ethernet Adapter-> Properties ->Advanced tab



3.1.4.3 Configuring SwitchX® Based Switch System

➢ ***To enable RoCE, the SwitchX should be configured as follows:***

- Ports facing the host should be configured as access ports, and either use global pause or Port Control Protocol (PCP) for priority flow control
- Ports facing the network should be configured as trunk ports, and use Port Control Protocol (PCP) for priority flow control

For further information on how to configure SwitchX, please refer to SwitchX User Manual.

3.1.4.4 Configuring Arista Switch

Step 1. Set the ports that face the hosts as trunk.

```
(config)# interface et10
(config-if-Et10)# switchport mode trunk
```

Step 2. Set VID allowed on trunk port to match the host VID.

```
(config-if-Et10)# switchport trunk allowed vlan 100
```

Step 3. Set the ports that face the network as trunk.

```
(config)# interface et20
(config-if-Et20)# switchport mode trunk
```

Step 4. Assign the relevant ports to LAG.

```
(config)# interface et10
(config-if-Et10)# dcbx mode ieee
(config-if-Et10)# speed forced 40gfull
(config-if-Et10)# channel-group 11 mode active
```

Step 5. Enable PFC on ports that face the network.

```
(config)# interface et20
(config-if-Et20)# load-interval 5
(config-if-Et20)# speed forced 40gfull
(config-if-Et20)# switchport trunk native vlan tag
(config-if-Et20)# switchport trunk allowed vlan 11
(config-if-Et20)# switchport mode trunk
(config-if-Et20)# dcbx mode ieee
(config-if-Et20)# priority-flow-control mode on
(config-if-Et20)# priority-flow-control priority 3 no-drop
```

3.1.4.4.1 Using Global Pause (Flow Control)

➢ ***To enable Global Pause on ports that face the hosts, perform the following:***

```
(config)# interface et10
(config-if-Et10)# flowcontrol receive on
(config-if-Et10)# flowcontrol send on
```

3.1.4.4.2 Using Priority Flow Control (PFC)

➢ ***To enable Global Pause on ports that face the hosts, perform the following:***

```
(config)# interface et10
(config-if-Et10)# dcbx mode ieee
```

```
(config-if-Et10)# priority-flow-control mode on
(config-if-Et10)# priority-flow-control priority 3 no-drop
```

3.1.4.5 Configuring Router (PFC only)

The router uses L3's DSCP value to mark the egress traffic of L2 PCP. The required mapping, maps the three most significant bits of the DSCP into the PCP. This is the default behavior, and no additional configuration is required.

3.1.4.5.1 Copying Port Control Protocol (PCP) between Subnets

The captured PCP option from the Ethernet header of the incoming packet can be used to set the PCP bits on the outgoing Ethernet header.

3.1.4.6 Configuring the RoCE Mode

Configuring the RoCE mode requires the following:

- RoCE mode is configured per-driver and is enforced on all the devices in the system

The supported RoCE modes depend on the firmware installed. If the firmware does not support the needed mode, the fallback mode would be the maximum supported RoCE mode of the installed NIC.

RoCE mode can be enabled and disabled via PowerShell.

➢ ***To enable RoCE MAC Based using the PowerShell:***

- Open the PowerShell and run:

```
PS $ Set-MlnxDriverCoreSetting -RoceMode 1
```

➢ ***To enable RoCE IP Based to using the Powershell:***

```
PS $ Set-MlnxDriverCoreSetting -RoceMode 1.25
```

➢ ***To enable RoCE v2 using the PowerShell:***

- Open the PowerShell and run:

```
PS $ Set-MlnxDriverCoreSetting -RoceMode 2
```

➢ ***To disable any version of RoCE using the PowerShell:***

- Open the PowerShell and run:

```
PS $ Set-MlnxDriverCoreSetting -RoceMode 0
```

➢ ***To check current version of RoCE using the PowerShell:***

- Open the PowerShell and run:

```
PS $ Get-MlnxDriverCoreSetting
```

- Example output:

```
Caption               : DriverCoreSettingData 'mlx4_bus'
Description           : Mellanox Driver Option Settings
.
.
.
RoceMode              : 0
```

3.1.5 Load Balancing, Fail-Over (LBFO) and VLAN

Windows Server 2012 and above supports load balancing as part of the operating system. Please refer to Microsoft guide “NIC Teaming in Windows Server 2012” following the link below:

http://social.technet.microsoft.com/wiki/contents/articles/14951.nic-teaming-in-windows-server-2012.aspx

For other earlier operating systems, please refer to the sections below.

3.1.5.1 Adapter Teaming

Adapter teaming can group a group of ports inside a network adapter or a number of physical network adapters into virtual adapters that provide the fault-tolerance and load-balancing functions. Depending on the teaming mode, one or more interfaces can be active. The non-active interfaces in a team are in a standby mode and will take over the network traffic in the event of a link failure in the active interfaces. All of the active interfaces in a team participate in load-balancing operations by sending and receiving a portion of the total network traffic.

3.1.5.1.1 Teaming (Bundle) Modes

1. **Fault Tolerance**
   Provides automatic redundancy for the server’s network connection. If the primary adapter fails, the secondary adapter (currently in a standby mode) takes over. Fault Tolerance is the basis for each of the following teaming types and is inherent in all teaming modes.
2. **Switch Fault Tolerance**
   Provides a failover relationship between two adapters when each adapter is connected to a separate switch.
3. **Send Load Balancing**
   Provides load balancing of transmit traffic and fault tolerance. The load balancing performs only on the send port.
4. **Load Balancing (Send & Receive)**
   Provides load balancing of transmit and receive traffic and fault tolerance. The load balancing splits the transmit and receive traffic statically among the team adapters (without changing the base of the traffic loading) based on the source/destination MAC and IP addresses.
5. **Adaptive Load Balancing**
   The same functionality as Load Balancing (Send & Receive). In case of traffic load in one of the adapters, the load balancing channels the traffic between the other team adapter.
6. **Dynamic Link Aggregation (802.3ad)**
   Provides dynamic link aggregation allowing creation of one or more channel groups using same speed or mixed-speed server adapters.
7. **Static Link Aggregation (802.3ad)**
   Provides increased transmission and reception throughput in a team comprised of two to eight adapter ports through static configuration.

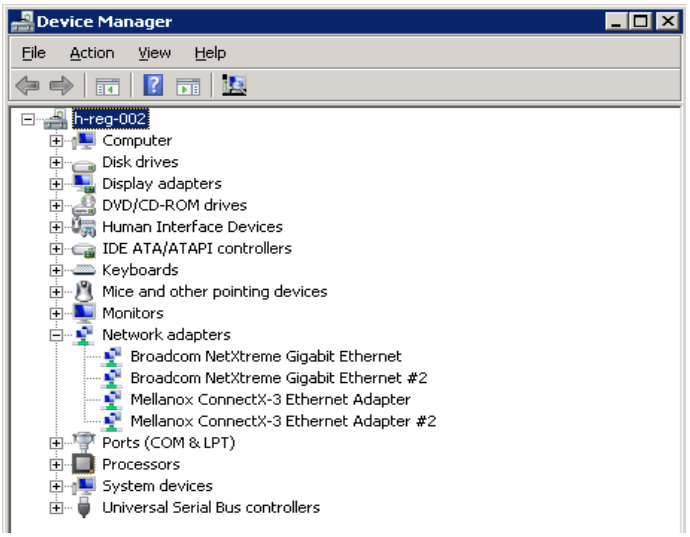
   If the switch connected to the HCA supports 802.3ad the recommended setting is teaming mode 6.

3.1.5.2 Creating a Load Balancing and Fail-Over (LBFO) Bundle

LBFO is used to balance the workload of packet transfers by distributing the workload over a bundle of network instances and to set a secondary network instance to take over packet indications and information requests if the primary network instance fails.

The following steps describe the process of creating an LBFO bundle.

Step 1. Display the Device Manager.



Step 2. Right-click a Mellanox ConnectX 10Gb Ethernet adapter (under "Network adapters" list) and left click Properties. Select the LBFO tab from the Properties window.

It is not recommended to open the Properties window of more than one adapter simultaneously.

The LBFO dialog enables creating, modifying or removing a bundle.

Only Mellanox Technologies adapters can be part of the LBFO.

➢ ***To create a new bundle, perform the following***

Step 1. Click Create.

Step 2. Enter a (unique) bundle name.

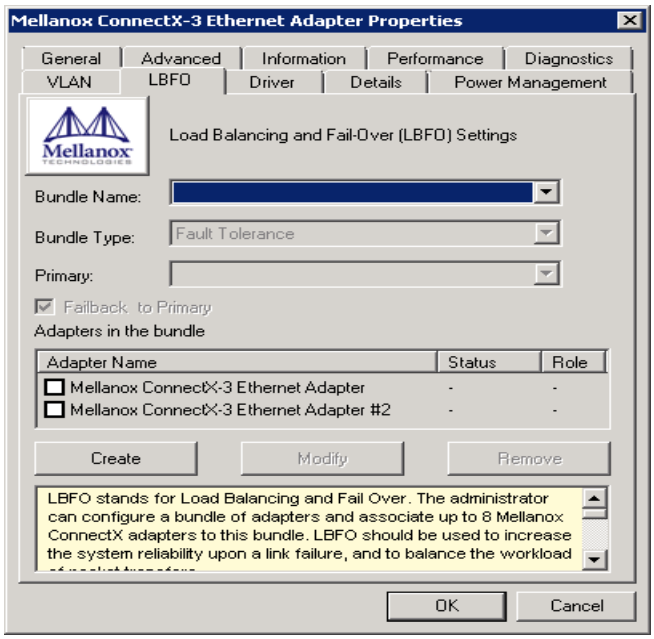
Step 3. Select a bundle type.

Step 4. Select the adapters to be included in the bundle (that have not been associated with a VLAN).

Step 5. [Optional] Select Primary Adapter.
An active-passive scenario used for data transfer of link disconnecting. In such scenario, the system uses one of the other interfaces. When the primary link comes up, the LBFO interface returns to transfer data using the primary interface. If the primary adapter is not selected, the primary interface is selected randomly.

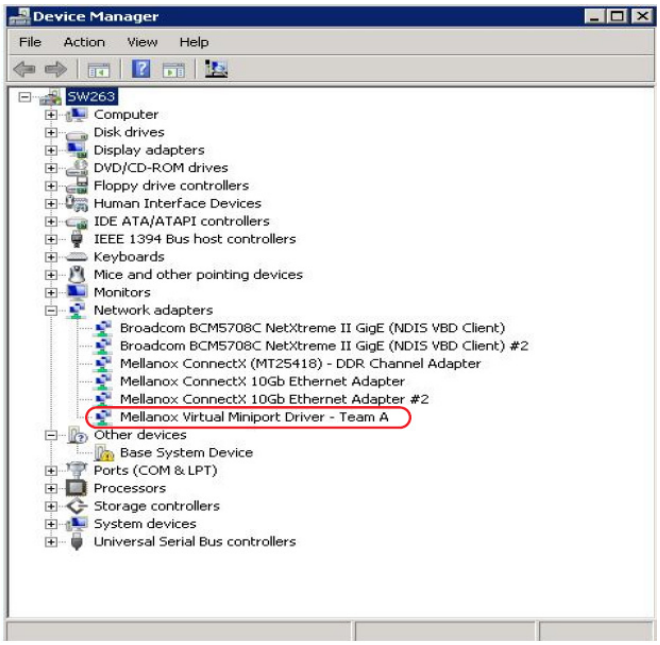
Step 6. [Optional] Failback to Primary

Step 7. Check the checkbox.



The newly created virtual Mellanox adapter representing the bundle will be displayed by the Device Manager under "Network adapters" in the following format (see the figure below):

```
Mellanox Virtual Miniport Driver - Team <bundle_name>
```



➢ ***To modify an existing bundle, perform the following:***

a. Select the desired bundle and click Modify

b. Modify the bundle name, its type, and/or the participating adapters in the bundle

c. Click the Commit button

➢ ***To remove an existing bundle, select the desired bundle and click Remove. You will be prompted to approve this action.***

Notes on this step:

a. Each adapter that participates in a bundle has two properties:
   - Status: Connected/Disconnected/Disabled
   - Role: Active or Backup

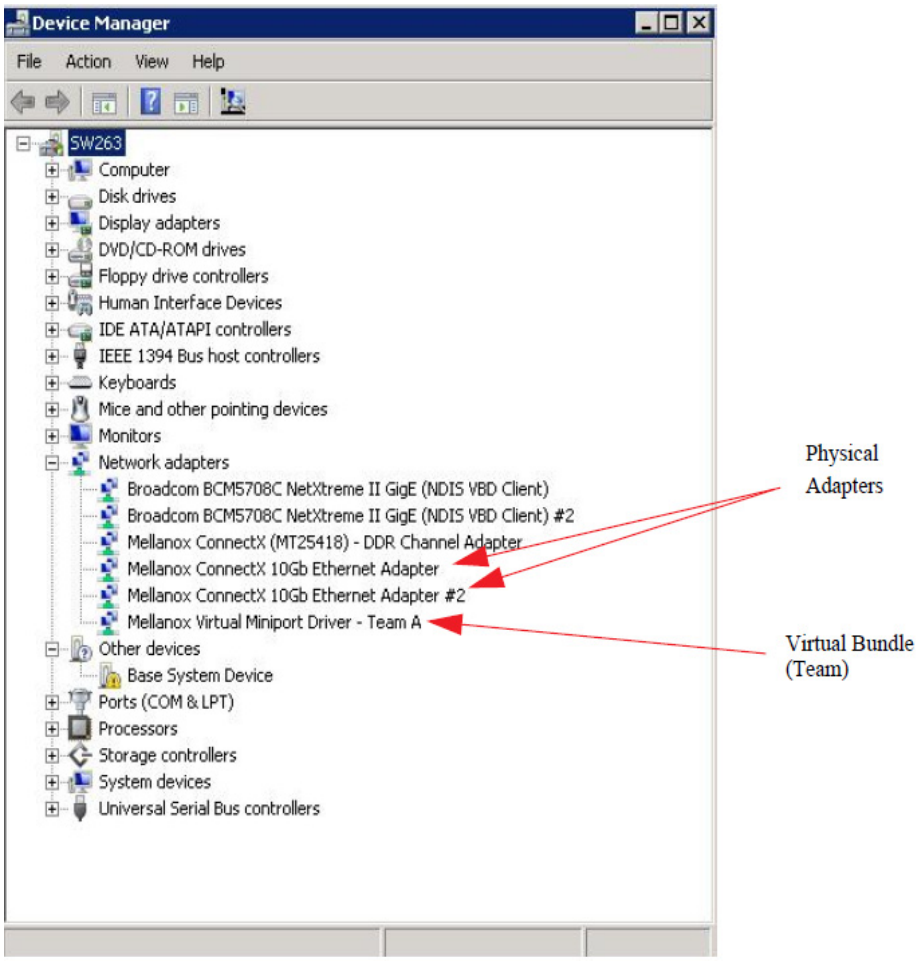
b. Each network adapter that is added or removed from a bundle gets refreshed (i.e. disabled then enabled). This may cause a temporary loss of connection to the adapter.

c. In case a bundle loses one or more network adapters by a "create" or "modify" operation, the remaining adapters in the bundle are automatically notified of the change.

3.1.5.3 Creating a Port VLAN in Windows 2008 R2

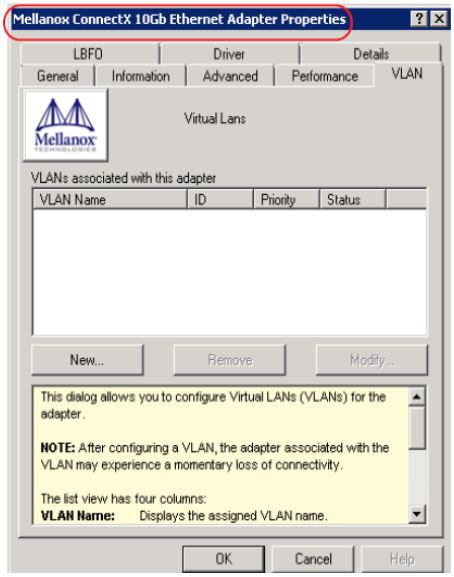
You can create a Port VLAN either on a physical Mellanox ConnectX® EN adapter or a virtual bundle (team). The following steps describe how to create a port VLAN.

Step 1. Display the Device Manager.

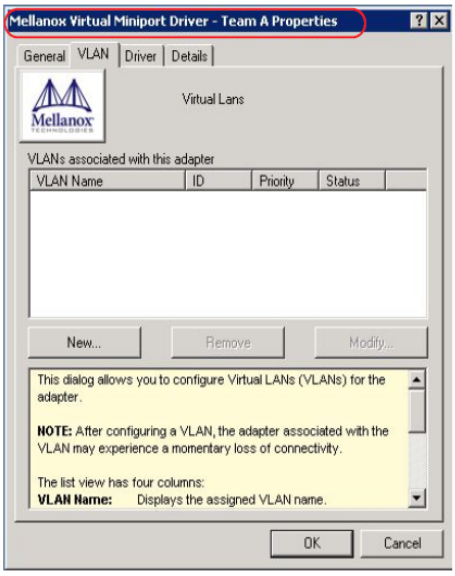


Step 2. Right-click a Mellanox network adapter (under "Network adapters" list) and left-click Properties. Select the VLAN tab from the Properties sheet.

Physical Adapter

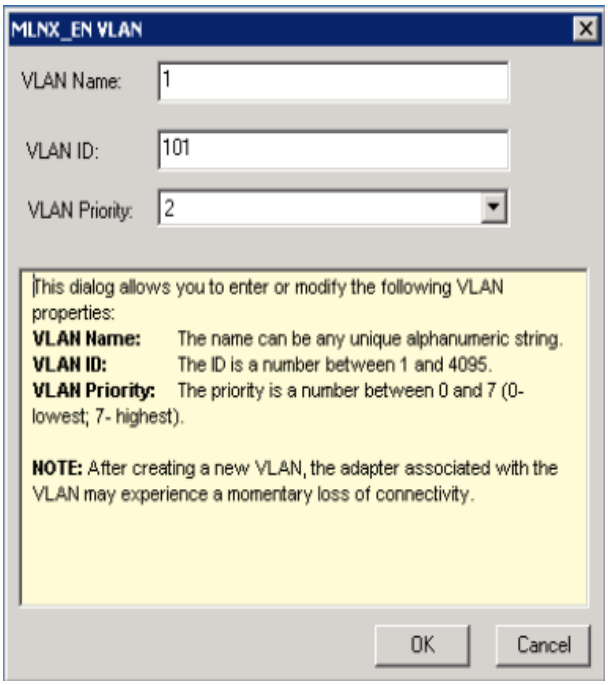


Virtual Bundle (Team)



If a physical adapter has been added to a bundle (team), the VLAN tab will not be displayed.

Step 3. Click New to open a VLAN dialog window. Enter the desired VLAN Name and VLAN ID, and select the VLAN Priority.

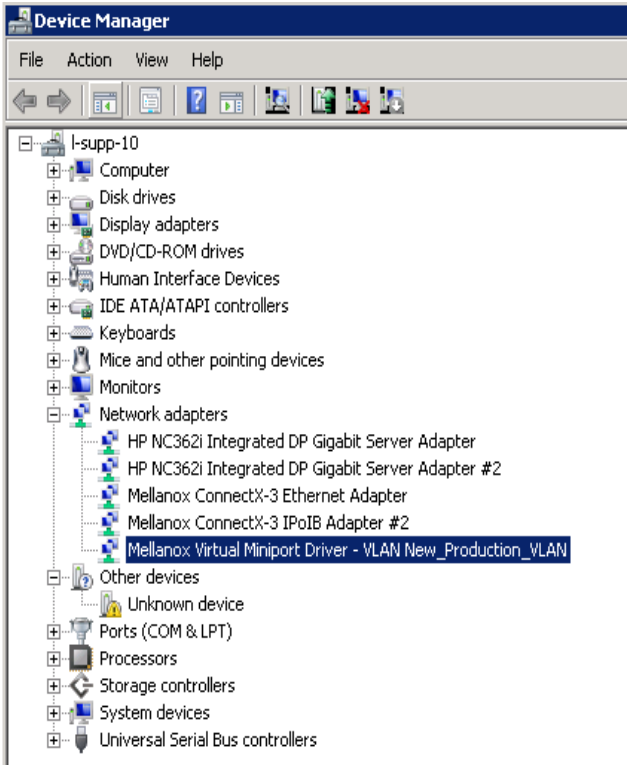


After installing the first virtual adapter (VLAN) on a specific port, the port becomes disabled. This means that it is not possible to bind to this port until all the virtual adapters associated with it are removed.

When using a VLAN, the network address is configured using the VLAN ID. Therefore, the VLAN ID on both ends of the connection must be the same.

Step 4. Verify the new VLAN(s) by opening the Device Manager window or the Network Connections window. The newly created VLAN will be displayed in the following format.

```
Mellanox Virtual Miniport Driver - VLAN <name>
```



3.1.5.4 Removing a Port VLAN in Windows 2008 R2

➢ ***To remove a port VLAN, perform the following steps:***

Step 1. In the Device Manager window, right-click the network adapter from which the port VLAN was created.

Step 2. Left-click Properties.

Step 3. Select the VLAN tab from the Properties sheet.

Step 4. Select the VLAN to be removed.

Step 5. Click Remove and confirm the operation.

3.1.5.5 Configuring a Port to Work with VLAN in Windows 2012 and Above

In this procedure you DO NOT create a VLAN, rather use an existing VLAN ID.

➢ ***To configure a port to work with VLAN using the Device Manager.***

Step 1. Open the Device Manager.

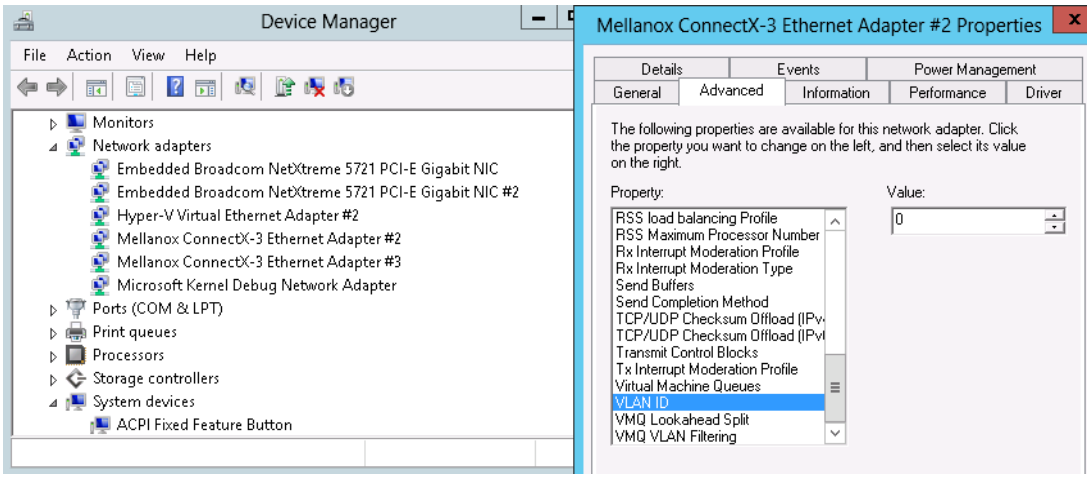
Step 2. Go to the Network adapters.

Step 3. Right click Properties on Mellanox ConnectX®-3 Ethernet Adapter card.

Step 4. Go to Advanced tab.

Step 5. Choose the VLAN ID in the Property window.

Step 6. Set its value in the Value window.



3.1.6 Header Data Split

The header-data split feature improves network performance by splitting the headers and data in received Ethernet frames into separate buffers. The feature is disabled by default and can be enabled in the Advanced tab (Performance Options) from the Properties window.

For further information, please refer to the MSDN library:

http://msdn.microsoft.com/en-us/library/windows/hardware/ff553723(v=VS.85).aspx

3.1.7 Ports TX Arbitration

On a setup with a dual-port NIC with both ports at link speed of 40GbE, each individual port can achieve maximum line rate. When both ports are running simultaneously in a high throughput scenario, the total throughput is bottlenecked by the PCIe bus, and in this case each port may not achieve its maximum of 40GbE.

Ports TX Arbitration ensures bandwidth precedence is given to one of the ports on a dual-port NIC, enabling the preferred port to achieve the maximum throughput and the other port taking up the rest of the remaining bandwidth.

➢ ***To configure Ports TX Arbitration:***

Step 1. Open the Device Manager.

Step 2. Go to the Network adapters.

Step 3. Right click ' Properties on Mellanox ConnectX®-3 Ethernet Adapter card.

Step 4. Go to Advanced tab.

Step 5. Choose the `'Tx Throughput Port Arbiter'` option.

Step 6. Set one of the following values:

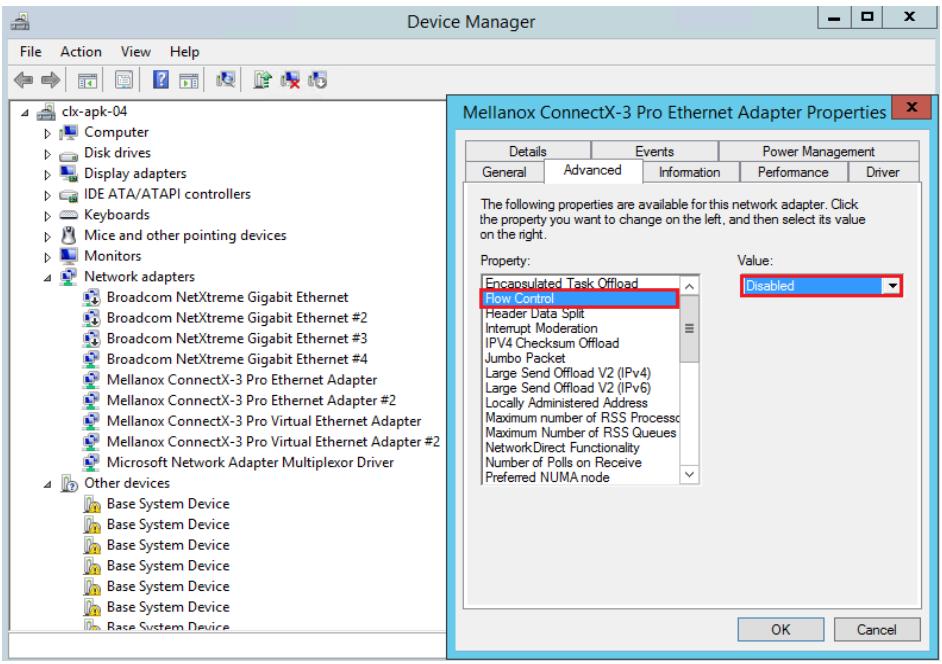
- Best Effort (Default) - Default behavior. No precedence is given to this port over the other.
- Guaranteed - Give higher precedence to this port.
- Not Present - No configuration exists, defaults are used.

3.1.8 Configuring Quality of Service (QoS)

Prior to configuring Quality of Service, you must install Data Center Bridging using one of the following methods:

➢ ***To Disable Flow Control Configuration***

Device manager->Network adapters->Mellanox ConnectX-3 Ethernet Adapter->Properties->Advanced tab



➢ ***To install the Data Center Bridging using the Server Manager:***

Step 1. Open the 'Server Manager'.

Step 2. Select 'Add Roles and Features'.

Step 3. Click Next.

Step 4. Select 'Features' on the left panel

Step 5. Check the 'Data Center Bridging' checkbox.

Step 6. Click 'Install'.

➢ ***To install the Data Center Bridging using PowerShell:***

Step 1. Enable Data Center Bridging (DCB).

```
PS $ Install-WindowsFeature Data-Center-Bridging
```

➢ ***To configure QoS on the host:***

The procedure below is not saved after you reboot your system. Hence, we recommend you create a script using the steps below and run it on the local machine. Please see the procedure below on how to add the script to the local machine startup scripts.

Step 1. Change the Windows PowerShell execution policy.

```
PS $ Set-ExecutionPolicy AllSigned
```

Step 2. Remove the entire previous QoS configuration.

```
PS $ Remove-NetQosTrafficClass
PS $ Remove-NetQosPolicy -Confirm:$False
```

Step 3. Set the DCBX Willing parameter to false as Mellanox drivers do not support this feature.

```
PS $ set-NetQosDcbxSetting -Willing 0
```

Step 4. Create a Quality of Service (QoS) policy and tag each type of traffic with the relevant priority.

In this example we used TCP/UDP priority 1, ND/NDK priority 3.

```
PS $ New-NetQosPolicy "SMB" -store Activestore -NetDirectPortMatchCondition 445 -
PriorityValue8021Action 3
PS $ New-NetQosPolicy "DEFAULT" -store Activestore -Default -PriorityValue8021Action 3
PS $ New-NetQosPolicy "TCP" -store Activestore -IPProtocolMatchCondition TCP -
PriorityValue8021Action 1
PS $ New-NetQosPolicy "UDP" -store Activestore -IPProtocolMatchCondition UDP -
PriorityValue8021Action 1
```

Step 5. [Optional] If VLANs are used, mark the egress traffic with the relevant VlanID.
The NIC is referred as "Ethernet 4" in the examples below.

```
PS $ Set-NetAdapterAdvancedProperty -Name "Ethernet 4" -RegistryKeyword "VlanID" -Reg-
istryValue "55"
```

Step 6. [Optional] Configure the IP address for the NIC.

If DHCP is used, the IP address will be assigned automatically.

```
PS $ Set-NetIPInterface -InterfaceAlias "Ethernet 4" -DHCP Disabled
```

```
PS $ Remove-NetIPAddress -InterfaceAlias "Ethernet 4" -AddressFamily IPv4 -Con-
firm:$false
PS $ New-NetIPAddress -InterfaceAlias "Ethernet 4" -IPAddress 192.168.1.10 -Prefix-
Length 24 -Type Unicast
```

Step 7. [Optional] Set the DNS server (assuming its IP address is 192.168.1.2).

```
PS $ Set-DnsClientServerAddress -InterfaceAlias "Ethernet 4" -ServerAddresses
192.168.1.2
```

After establishing the priorities of ND/NDK traffic, the priorities must have PFC enabled on them.

Step 8. Disable Priority Flow Control (PFC) for all other priorities except for 3.

```
PS $ Disable-NetQosFlowControl 0,1,2,4,5,6,7
```

Step 9. Enable QoS on the relevant interface.

```
PS $ Enable-NetAdapterQos -InterfaceAlias "Ethernet 4"
```

Step 10. Enable PFC on priority 3.

```
PS $ Enable-NetQosFlowControl -Priority 3
```

Step 11. Configure Priority 3 to use ETS.

```
PS $ New-NetQosTrafficClass -name "SMB class" -priority 3 -bandwidthPercentage 50 -
Algorithm ETS
```

➢ ***To add the script to the local machine startup scripts:***

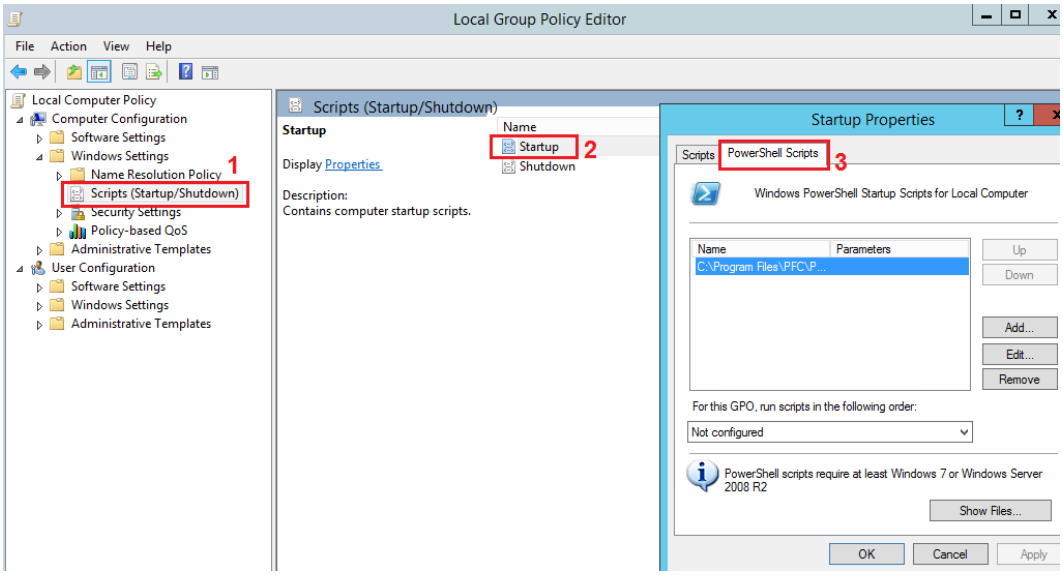
Step 1. From the PowerShell invoke.

```
gpedit.msc
```

Step 2. In the pop-up window, under the 'Computer Configuration' section, perform the following:

1. Select Windows Settings
2. Select Scripts (Startup/Shutdown)
3. Double click Startup to open the Startup Properties

4. Move to “PowerShell Scripts” tab



5. Click Add

   The script should include only the following commands:

```
PS $ Remove-NetQosTrafficClass
PS $ Remove-NetQosPolicy -Confirm:$False
PS $ set-NetQosDcbxSetting -Willing 0
PS $ New-NetQosPolicy "SMB" -Policystore Activestore -NetDirectPortMatchCondition 445 -
PriorityValue8021Action 3
PS $ New-NetQosPolicy "DEFAULT" -Policystore Activestore -Default -
PriorityValue8021Action 3
PS $ New-NetQosPolicy "TCP" -Policystore Activestore -IPProtocolMatchCondition TCP -
PriorityValue8021Action 1
PS $ New-NetQosPolicy "UDP" -Policystore Activestore -IPProtocolMatchCondition UDP -
PriorityValue8021Action 1
PS $ Disable-NetQosFlowControl 0,1,2,4,5,6,7
PS $ Enable-NetAdapterQos -InterfaceAlias "port1"
PS $ Enable-NetAdapterQos -InterfaceAlias "port2"
PS $ Enable-NetQosFlowControl -Priority 3
PS $ New-NetQosTrafficClass -name "SMB class" -priority 3 -bandwidthPercentage 50 -
Algorithm ETS
```

6. Browse for the script's location.
7. Click OK
8. To confirm the settings applied after boot run:

```
PS $ get-netqospolicy -policystore activestore
```

3.1.8.1 Enhanced Transmission Selection

Enhanced Transmission Selection (ETS) provides a common management framework for assignment of bandwidth to frame priorities as described in the IEEE 802.1Qaz specification:

http://www.ieee802.org/1/files/public/docs2008/az-wadekar-ets-proposal-0608-v1.01.pdf

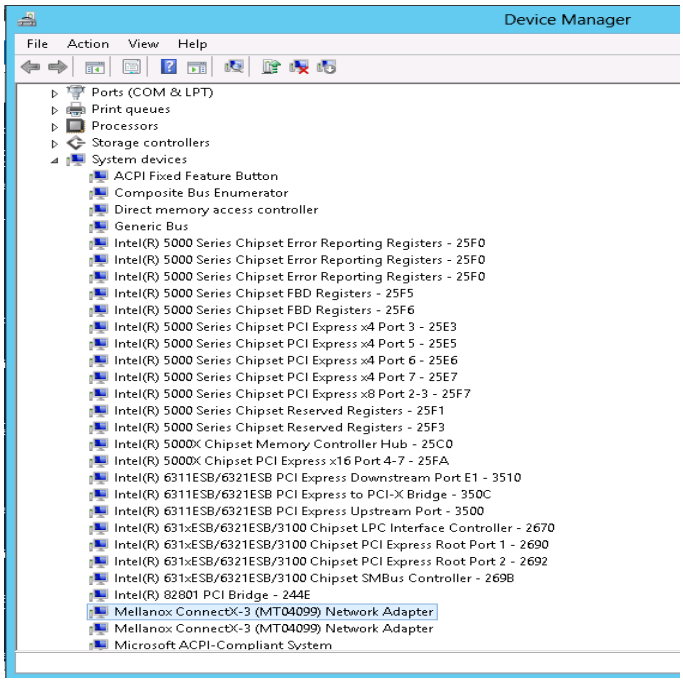
For further details on configuring ETS on Windows™ Server, please refer to:

http://technet.microsoft.com/en-us/library/hh967440.aspx

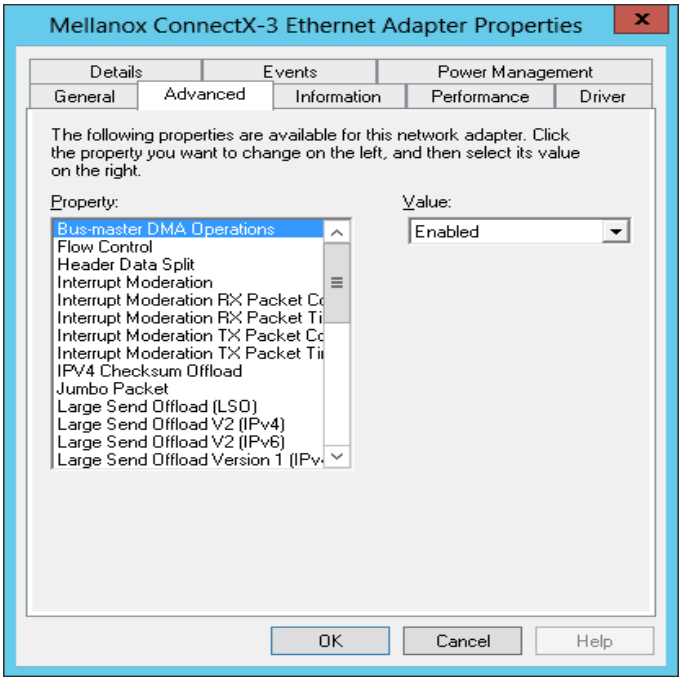
3.1.9 Configuring the Ethernet Driver

The following steps describe how to configure advanced features.

Step 1. Display the Device Manager.



Step 2. Right-click a Mellanox network adapter (under "Network adapters" list) and left-click Properties. Select the Advanced tab from the Properties sheet.



Step 3. Modify configuration parameters to suit your system.

Please note the following:

a. For help on a specific parameter/option, check the help button at the bottom of the dialog.

b. If you select one of the entries Off-load Options, Performance Options, or Flow Control Options, you'll need to click the Properties button to modify parameters via a pop-up dialog.

3.1.10 Differentiated Services Code Point (DSCP)

DSCP is a mechanism used for classifying network traffic on IP networks. It uses the 6-bit Differentiated Services Field (DS or DSCP field) in the IP header for packet classification purposes. Using Layer 3 classification enables you to maintain the same classification semantics beyond local network, across routers.

Every transmitted packet holds the information allowing network devices to map the packet to the appropriate 802.1Qbb CoS. For DSCP based PFC the packet is marked with a DSCP value in the Differentiated Services (DS) field of the IP header.

3.1.10.1 Setting the DSCP in the IP Header

Marking DSCP value in the IP header is done differently for IP packets constructed by the NIC (e.g. RDMA traffic) and for packets constructed by the IP stack (e.g. TCP traffic).

- For IP packets generated by the IP stack, the DSCP value is provided by the IP stack. The NIC does not validate the match between DSCP and Class of Service (CoS) values. CoS and DSCP values are expected to be set through standard tools, such as PowerShell command New-NetQosPolicy using PriorityValue8021Action and DSCPAction flags respectively.
- For IP packets generated by the NIC (RDMA), the DSCP value is generated according to the CoS value programmed for the interface. CoS value is set through standard tools, such as PowerShell command New-NetQosPolicy using PriorityValue8021Action flag. The NIC uses a mapping table between the CoS value and the DSCP value configured through the RroceDscpMarkPriorityFlow- Control[0-7] Registry keys

3.1.10.2 Configuring Quality of Service for TCP and RDMA Traffic

Step 1. Verify that DCB is installed and enabled (is not installed by default).

```
PS $ Install-WindowsFeature Data-Center-Bridging
```

Step 2. Import the PowerShell modules that are required to configure DCB.

```
PS $ import-module NetQos
PS $ import-module DcbQos
PS $ import-module NetAdapter
```

Step 3. Configure DCB.

```
PS $ Set-NetQosDcbxSetting -Willing 0
```

Step 4. Enable Network Adapter QoS.

```
PS $ Set-NetAdapterQos -Name "Cx3Pro_ETH_P1" -Enabled 1
```

Step 5. Enable Priority Flow Control (PFC) on the specific priority 3,5.

```
PS $ Enable-NetQosFlowControl 3,5
```

3.1.10.3 Configuring DSCP for TCP Traffic

- Create a QoS policy to tag All TCP/UDP traffic with CoS value 1 and DSCP value 9.

```
PS $ New-NetQosPolicy "DEFAULT" -PriorityValue8021Action 3 -DSCPAction 9
```

DSCP can also be configured per protocol.

```
PS $ New-NetQosPolicy "TCP"  -IPProtocolMatchCondition TCP -PriorityValue8021Action 3 -
DSCPAction 16
PS $ New-NetQosPolicy "UDP"  -IPProtocolMatchCondition UDP -PriorityValue8021Action 3 -
DSCPAction 32
```

3.1.10.4 Configuring DSCP for RDMA Traffic

- Create a QoS policy to tag the ND traffic for port 10000 with CoS value 3.

```
PS $ New-NetQosPolicy "ND10000" -NetDirectPortMatchCondition 10000 -
PriorityValue8021Action 3
```

Related Commands:

- Get-NetAdapterQos - Gets the QoS properties of the network adapter
- Get-NetQosPolicy - Retrieves network QoS policies
- Get-NetQosFlowControl - Gets QoS status per priority

3.1.10.5 Registry Settings

The following attributes must be set manually and will be added to the miniport registry.

Table 7 - DSCP Registry Keys Settings

Registry Key	Description
TxUntagPriorityTag	If 0x1, do not add 802.1Q tag to transmitted packets which are assigned 802.1p priority, but are not assigned a non-zero VLAN ID (i.e. priority-tagged). Default 0x0, for DSCP based PFC set to 0x1.
RxUntaggedMapToLossless	If 0x1, all untagged traffic is mapped to the lossless receive queue. Default 0x0, for DSCP based PFC set to 0x1.
RroceDscpMarkPriorityFlowControl_<ID>	A value to mark DSCP for RoCE v2 packets assigned to CoS=ID, when priority flow control is enabled. The valid values range is from 0 to 63, Default is ID value, e.g. RroceDscpMarkPriorityFlowControl_3 is 3. ID values range from 0 to 7.

For changes to take affect, please restart the network adapter after changing this registry key.

3.1.10.5.1Default Settings

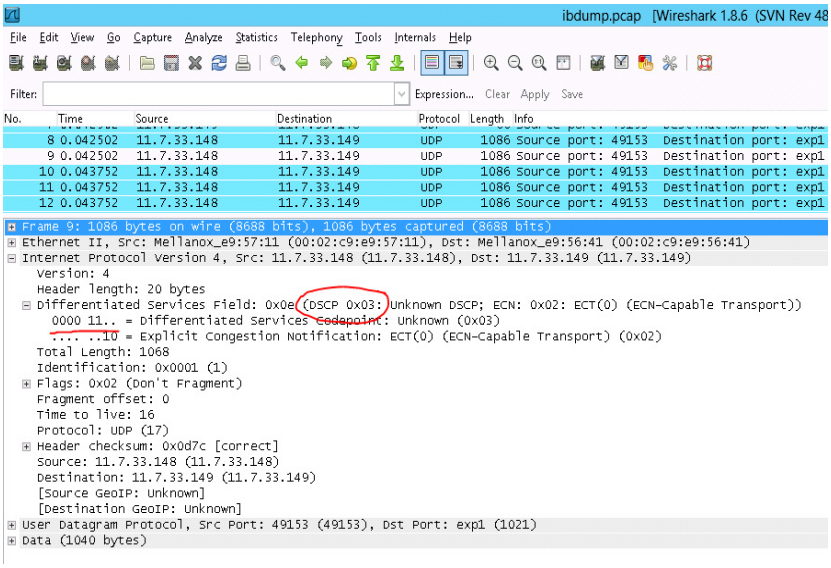
When DSCP configuration registry keys are missing in the miniport registry, the following defaults are assigned.

Table 8 - DSCP Default Registry Keys Settings

Registry Key	Default Value
TxUntagPriorityTag	0
RxUntaggedMapToLossles	0
RroceDscpMarkPriorityFlowControl_0	0
RroceDscpMarkPriorityFlowControl_1	1
RroceDscpMarkPriorityFlowControl_2	2
RroceDscpMarkPriorityFlowControl_3	3
RroceDscpMarkPriorityFlowControl_4	4
RroceDscpMarkPriorityFlowControl_5	5
RroceDscpMarkPriorityFlowControl_6	6
RroceDscpMarkPriorityFlowControl_7	7

3.1.10.6 DSCP Sanity Testing

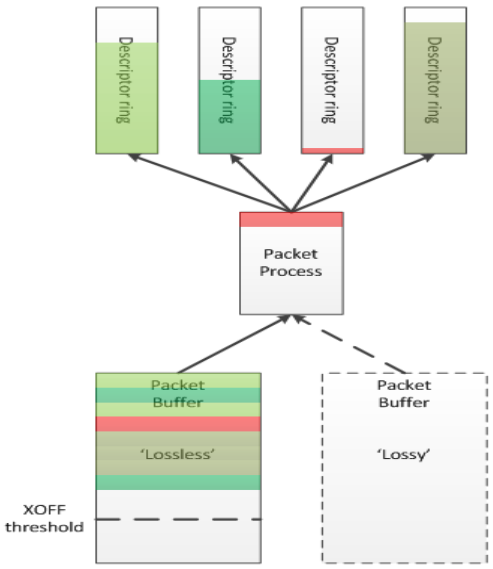
To verify that all QoS and DSCP settings were correct, you can capture incoming and outgoing traffic by using the ibdump tool and see the DSCP value in the captured packets as displayed in the figure below.



3.1.11 Lossless TCP

Inbound packets are stored in the data buffers. They are split into 'Lossy' and 'Lossless' according to the priority field in the 802.1Q VLAN tag. In DSCP based PFC, all traffic is directed to the 'Lossless' buffer. Packets are taken out of the packet buffer in the same order they were stored, and moved into processing, where a destination descriptor ring is selected. The packet is then scattered into the appropriate memory buffer, pointed by the first free descriptor.

Figure 4: Lossless TCP



When the 'Lossless' packet buffer crosses the XOFF threshold, the adapter sends 802.3x pause frames according to the port configuration: Global pause, or per-priority 802.1Qbb pause (PFC), where only the priorities configured as 'Lossless' will be noted in the pause frame. Packets arriving while the buffer is full are dropped immediately.

During packet processing, if the selected descriptor ring has no free descriptors, two modes for handling are available:

3.1.11.1 Drop Mode

In this mode, a packet arriving to a descriptor ring with no free descriptors is dropped, after verifying that there are really no free descriptors. This allows isolation of the host driver execution delays from the network, as well as isolation between different SW entities sharing the adapter (e.g. SR-IOV VMs).

3.1.11.2 Poll Mode

In this mode, a packet arriving to a descriptor ring with no free descriptors will patiently wait until a free descriptor is posted. All processing for this packet and the following packets is halted, while free descriptor status is polled. This behavior will propagate the backpressure into the Rx buffer which will accumulate incoming packets. When XOFF threshold is crossed, Flow Control mechanisms mentioned earlier will stop the remote transmitters, thus avoiding packets from being dropped.

Since this mode breaks the aforementioned isolation, the adapter offers a mitigation mechanism that limits the amount of time a packet may wait for a free descriptor, while halting all packet processing. When the allowed time expires the adapter reverts to the 'Drop Mode' behavior.

3.1.11.3 Default behavior

By default the adapter works in 'Drop Mode'. The adapter reverts to this mode upon initialization/ restart.

3.1.11.4 Known Limitations

- The feature is not available for SR-IOV Virtual Functions
- It is recommended that the feature be used only when the port is configured to maintain flow control.
- It is recommended not to exceed typical timeout values of management protocols, usually in the order of several seconds.
- In order for the feature to effectively prevent packet drops, the DPC load duration needs to be lower than the TCP retransmission timeout.
- The feature is only activated if neither of the ports is IB.

3.1.11.5 System Requirements

- Operating System: Windows 2012 or Windows 2012 R2
- Firmware: 2.31.5050

3.1.11.6 Enabling/Disabling Lossless TCP

This feature is controlled using the registry key `DelayDropTimeout` that enables Lossless TCP capability in hardware and by Set OID `OID_MLX_DROPLESS_MODE` which triggers transition to/from Lossless (poll) mode.

3.1.11.6.1 Enabling Lossless TCP Using The Registry Key `DelayDropTimeout`:

Registry Key location:

```
HKLM\SYSTEM\CurrentControlSet\Control\Class\Class\{4d36e972-
e325-11ce-bfc1-08002be10318}\<nn>\DelayDropTimeout
```

For instructions on how to find interface index in registry <nn>, Please refer to Section 3.6.2, "Finding the Index Value of the Network Interface", on page 89

Key Name	Key Type	Values	Description
Delay-Drop-Timeout	REG_DWORD	• 0= disabled (default) • 1-65535=enabled	Choosing values between 1-65534 enables the feature, but the chosen value limits the amount of time a packet may wait for a free descriptor. The value is in units of 100 microseconds with inaccuracy of up to 2 units. The chosen time ranges between 100 microseconds and ~6.5 seconds. For example, DelayDropTimeout=3000 limits the wait time to 300 miliseconds (+/- 200 microseconds) Choosing the value of 65535 enables the feature but the amount of time a packet may wait for a free descriptor is infinite. **Note**: Changing the value of the DelayDropTimeout registry key requires restart of the network interface

3.1.11.6.2 Entering/Exiting Lossless Mode Using Set OID OID_MLX_DROPLESS_MODE:

In order to enter poll mode, registry value of DelayDropTimeout should be non-zero and OID_MLX_DROPLESS_MODE Set OID should be called with Information Buffer containing 1.

- OID_MLX_DROPLESS_MODE value: 0xFFA0C932
- OID Information Buffer Size: 1 byte
- OID Information Buffer Contents: 0 - exit poll mode; 1 - enter poll mode

3.1.11.7 Monitoring Lossless TCP State

In order to allow state transition monitoring, events are written to event log with mlx4_bus as the source. The associated events are listed in Table 9.

Table 9 - Lossless TCP Associated Events

Event ID	Event Description
0x0057 <Device Name>	Dropless mode entered on port <X>. Packets will not be dropped.
0x0058 <Device Name>	Dropless mode exited on port <X>. Drop mode entered; packets may now be dropped.
0x0059 <Device Name>	Delay drop timeout occurred on port <X>. Drop mode entered; packets may now be dropped.

3.1.12 Receive Side Scaling (RSS)

Mellanox WinOF Rev 4.80.50000 IPoIB and Ethernet drivers use NDIS 6.30 new RSS capabilities. The main changes are:

- Removed the previous limitation of 64 CPU cores
- Individual network adapter RSS configuration usage

RSS capabilities can be set per individual adapters as well as globally.

➢ ***To do so, set the registry keys listed below:***

For instructions on how to find interface index in registry <nn>, Please refer to Section 3.6.2, "Finding the Index Value of the Network Interface", on page 89.

Table 10 - Registry Keys Setting

Sub-key	Description
HKLM\SYSTEM\CurrentControlSet\Control\Class\{4d36e972-e325-11ce-bfc1-08002be10318}\<nn>*MaxRSSProcessors	**Maximum number of CPUs allotted.** Sets the desired maximum number of processors for each interface. The number can be different for each interface. Note: Restart the network adapter after you change this registry key.
HKLM\SYSTEM\CurrentControlSet\Control\Class\{4d36e972-e325-11ce-bfc1-08002be10318}\<nn>*RssBaseProcNumber	**Base CPU number.** Sets the desired base CPU number for each interface. The number can be different for each interface. This allows partitioning of CPUs across network adapters. Note: Restart the network adapter when you change this registry key.
HKLM\SYSTEM\CurrentControlSet\Control\Class\{4d36e972-e325-11ce-bfc1-08002be10318}\<nn>*NumaNodeID	**NUMA node affinitization**

Table 10 - Registry Keys Setting

Sub-key	Description
HKLM\SYSTEM\CurrentControlSet\Control\Class\{4d36e972-e325-11ce-bfc1-08002be10318}\<nn>*RssBaseProcGroup	Sets the RSS base processor group for systems with more than 64 processors.

3.2 Infiniband Network

3.2.1 Port Configuration

For more information on port configuration, please refer to 3.1.1 “Port Configuration,” on page 29.

3.2.2 OpenSM - Subnet Manager

OpenSM v3.3.11 is an InfiniBand Subnet Manager. In order to operate one host machine or more in the InfiniBand cluster., at least one Subnet Manger is required in the fabric.

Please use the embedded OpenSM in the WinOF package for testing purpose in small cluster. Otherwise, we recommend using OpenSM from FabricIT EFM™ or UFM® or MLNX-OS®.

OpenSM can run as a Windows service and can be started manually from the following directory: <installation_directory>\tools. OpenSM as a service will use the first active port, unless it receives a specific GUID.

OpenSM can be registered as a service from either the Command Line Interface (CLI) or the PowerShell.

The following are commands used from the CLI:

- ***To register it as a service execute the OpenSM service:***

```
> sc create OpenSM binPath= "c:\Program Files\Mella-
nox\MLNX_VPI\IB\Tools\opensm.exe
-service" start= auto
```

- ***To start OpenSM as a service:***

```
> sc start OpenSM
```

- ***To run OpenSM manually:***

```
> opensm.exe
```

For additional run options, enter: `"opensm.exe -h"`

The following are commands used from the PowerShell:

- ***To register it as a service execute the OpenSM service:***

```
> New-Service -Name "OpenSM" -BinaryPathName "`"C:\Program Files\Mella-
nox\MLNX_VPI\IB\Tools\opensm.exe`" --service -L 128" -DisplayName
"OpenSM" -Description "OpenSM for IB subnet" -StartupType Automatic
```

➢ ***To start OpenSM as a service run:***

```
> Start-Service OpenSM1
```

Notes

- For long term running, please avoid using the '-v' (verbosity) option to avoid exceeding disk quota.
- Running OpenSM on multiple servers may lead to incorrect OpenSM behavior.

Please do not run more than two instances of OpenSM in the subnet.

3.3 Upper Layer Protocols

3.3.1 IP over InfiniBand (IPoIB)

3.3.1.1 Modifying IPoIB Configuration

➢ ***To modify the IPoIB configuration after installation, perform the following steps:***

Step 1. Open Device Manager and expand Network Adapters in the device display pane.

Step 2. Right-click the Mellanox IPoIB Adapter entry and left-click Properties.

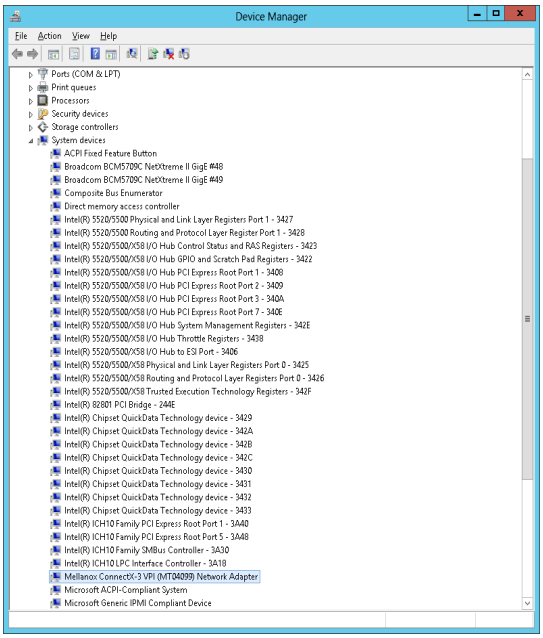
Step 3. Click the Advanced tab and modify the desired properties.

The IPoIB network interface is automatically restarted once you finish modifying IPoIB parameters. Consequently, it might affect any running traffic.

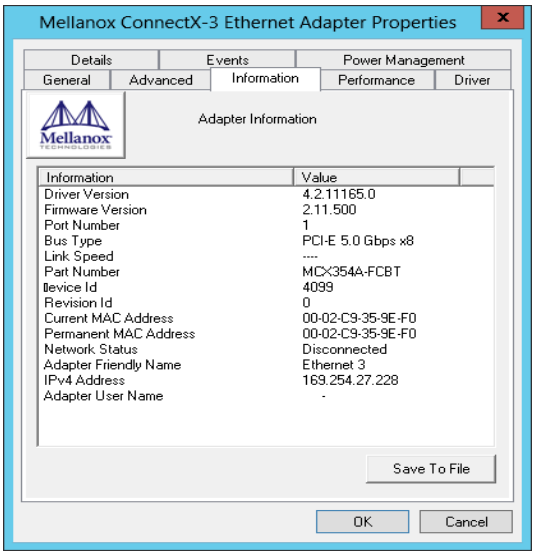
3.3.1.2 Displaying Adapter Related Information

To display a summary of network adapter software, firmware- and hardware-related information such as driver version, firmware version, bus interface, adapter identity, and network port link information, perform the following steps:

Step 1. Display the Device Manager.



Step 2. Select the Information tab from the Properties sheet.



To save this information for debug purposes, click **Save to File** and provide the output file name.

3.3.1.3 Assigning Port IP After Installation

For more information on port configuration, please refer to 3.3.1.3 “Assigning Port IP After Installation,” on page 62.

3.3.1.4 Receive Side Scaling (RSS)

For more information on port configuration, please refer to 3.1.12 “Receive Side Scaling (RSS),” on page 59.

3.3.1.5 Multiple Interfaces over non-default PKeys Support

OpenSM enables the configuration of partitions (PKeys) in an InfiniBand fabric. IPoIB supports the creation of multiple interfaces via the `part_man` tool. Each of those interfaces can be configured to use a different partition from the ones that were configured for OpenSM. This can allow partitioning of the IPoIB traffic between the different virtual IPoIB interfaces.

➢ ***To create a new interface on a new PKey on a native Windows machine:***

Step 1. Configure OpenSM to recognize the partition you would like to add.

For further details please refer to section 8.4 “Partitions” in MLNX OFED User Manual.

Step 2. Create a new interface using the part_man tool.

For further details please refer to section 4.2 “part_man - Virtual IPoIB Port Creation Utility,” on page 128.

Step 3. Assign Port IPs to the new interfaces.

For further details please refer to 3.1.2 “Assigning Port IP After Installation,” on page 30

Make sure the OpenSM is using the partitions configuration, and the new interfaces were configured to run over the same physical port.

➢ ***To create a new interface on a new PKey on a Windows virtual machine over a Linux host:***

On the Linux host:

Step 1. Configure the OpenSM to recognize the partition you would like to add.

For further details please refer to section 8.4 “Partitions” in MLNX OFED User Manual.

Step 2. Map the physical PKey table to the virtual PKey table used by the VM.

For further details please refer to section 4.15.6.2.4 “Partitioning IPoIB Communication using PKeys” in MLNX OFED User Manual.

On the Windows VM:

Step 1. Create a new interface using the `part_man` tool.

For further details please refer to section 4.2 “part_man - Virtual IPoIB Port Creation Utility,” on page 128.

Step 2. Assign Port IPs to the new interfaces.

For further details please refer to 3.1.2 “Assigning Port IP After Installation,” on page 30

Make sure the OpenSM using the partitions configuration, the physical-to-virtual PKey table mapping and the new interfaces were all configured over the same physical port.

➢ ***To assign a non-default PKey to the physical IPoIB port on a Windows virtual machine over a Linux host:***

On the Windows VM:

Step 1. Disable the driver on the port or disable the bus driver with all the ports it carries through the device manger.

On the Linux host:

Step 2. Configure the OpenSM to recognize the partition you would like to add.

For further details please refer to section 8.4 "Partitions" in MLNX OFED User Manual.

Step 3. Map the physical PKey table to the virtual PKey table used by the VM in the following way:

- Map the physical Pkey index you would like to use for the physical port to index 0 in the virtual Pkey table.
- Map the physical PKey index of the default PKey (index 0) to any index (for example: index1) in the virtual PKey table.

For further details please refer to section 4.15.6.2.4 "Partitioning IPoIB Communication using PKeys" in MLNX OFED User Manual.

On the Windows VM:

Step 4. Enable the drivers which were disabled.

Make sure the OpenSM using the partitions configuration, the physical-to-virtual PKey table mapping were configured over the same physical port.

➢ ***To change a configuration of an existing port:***

Step 1. Disable the driver on the port affected by the change you would like to make (or disable the bus driver with all the ports it carries) through the device manger in Windows OS.

Step 2. If required, configure the OpenSM to recognize the partition you would like to add or change. For further details please refer to section 8.4 "Partitions" in MLNX OFED User Manual.

Step 3. If the change is on a VM over a Linux host, map the physical PKey table to the virtual PKey table as required.
For further details please refer to section 4.15.6.2.4 "Partitioning IPoIB Communication using PKeys" in MLNX OFED User Manual.

Step 4. Enable the drivers you disabled in Windows OS.

3.4 Storage Protocols

3.4.1 Deploying Windows Server 2012 and Above with SMB Direct

The Server Message Block (SMB) protocol is a network file sharing protocol implemented in Microsoft Windows. The set of message packets that defines a particular version of the protocol is called a dialect.

The Microsoft SMB protocol is a client-server implementation and consists of a set of data packets, each containing a request sent by the client or a response sent by the server.

SMB protocol is used on top of the TCP/IP protocol or other network protocols. Using the SMB protocol allows applications to access files or other resources on a remote server, to read, create, and update them. In addition, it enables communication with any server program that is set up to receive an SMB client request.

3.4.1.1 Hardware and Software Prerequisites

The following are hardware and software prerequisites:

- Two or more machines running Windows Server 2012 and above
- One or more Mellanox ConnectX®-2, ConnectX®-3, or ConnectX®-3 Pro adapters for each server
- One or more Mellanox InfiniBand switches
- Two or more QSFP cables required for InfiniBand

3.4.1.2 SMB Configuration Verification

3.4.1.2.1 Verifying Network Adapter Configuration

Use the following PowerShell cmdlets to verify Network Direct is globally enabled and that you have NICs with the RDMA capability.

- Run on both the SMB server and the SMB client.

```
PS $ Get-NetOffloadGlobalSetting | Select NetworkDirect
PS $ Get-NetAdapterRDMA
PS $ Get-NetAdapterHardwareInfo
```

3.4.1.2.2 Verifying SMB Configuration

Use the following PowerShell cmdlets to verify SMB Multichannel is enabled, confirm the adapters are recognized by SMB and that their RDMA capability is properly identified.

- On the SMB client, run the following PowerShell cmdlets:

```
PS $ Get-SmbClientConfiguration | Select EnableMultichannel
PS $ Get-SmbClientNetworkInterface
```

- On the SMB server, run the following PowerShell cmdlets[1]:

```
PS $ Get-SmbServerConfiguration | Select EnableMultichannel
PS $ Get-SmbServerNetworkInterface
PS $ netstat.exe -xan | ? {$_ -match "445"}
```

3.4.1.2.3 Verifying SMB Connection

➢ ***To verify the SMB connection on the SMB client:***

Step 1. Copy the large file to create a new session with the SMB Server.

Step 2. Open a PowerShell window while the copy is ongoing.

Step 3. Verify the SMB Direct is working properly and that the correct SMB dialect is used.

```
PS $ Get-SmbConnection
PS $ Get-SmbMultichannelConnection
PS $ netstat.exe -xan | ? {$_ -match "445"}
```

1. The NETSTAT command confirms if the File Server is listening on the RDMA interfaces.

If you have no activity while you run the commands above, you might get an empty list due to session expiration and no current connections.

3.4.1.3 Verifying SMB Events that Confirm RDMA Connection

➢ ***To confirm RDMA connection, verify the SMB events:***

Step 1. Open a PowerShell window on the SMB client.

Step 2. Run the following cmdlets.
NOTE: Any RDMA-related connection errors will be displayed as well.

```
PS $ Get-WinEvent -LogName Microsoft-Windows-SMBClient/Operational | ? Message -match
"RDMA"
```

3.5 Virtualization

3.5.1 Virtual Ethernet Adapter

The Virtual Ethernet Adapter (VEA) provides a mechanism enabling multiple ethernet adapters on the same physical port. Each of these multiple adapters is referred to as a virtual ethernet adapter (VEA).

At present, one can have a total of two VEAs per port. The first VEA, normally the only adapter for the physical port, is referred to as a “physical VEA.” The second VEA, if present, is called a “virtual VEA”. currently only a single “Virtual VEA” is supported. The difference between a virtual and a physical VEA is that RDMA is only available through the physical VEA. In addition, certain settings for the port can only be configured on the physical VEA (see “VEA Feature Limitations” on page 67).

The VEA feature is designed to extend the OS capabilities and increase the usability of the network adapter. At present, once the user binds the RDMA capable network adapter to either teaming interface or Hyper-V, the RDMA capability (ND and NDK) is blocked by the OS. Hence if the user is interested to have RDMA and teaming or Hyper-V at the same time on the same physical Ethernet port, then he can take advantage of this feature: creating two VEAs the, first for RDMA and the second for the other use.

The user can manage VEAs using the “vea_man” tool. For further details on usage, please refer to “Vea_man- Virtual Ethernet” on page 130.

Virtual Ethernet Interfaces created by VEA_man are not tuned by the automatic performance tuning script, for optimal performance please follow the performance tuning guide and apply relevant changes to the VEA interface

3.5.1.1 System Requirements

- Operating System: Windows 2012 and Windows 2012 R2
- Firmware version: 2.31.5050 and above

3.5.1.2 VEA Feature Limitations

- RoCE (RDMA) is supported only on the physical VEA
- MTU (*JumboFrame registry key), QoS and, Flow Control are only configured from physical VEA
- No bandwidth allocation between the two interfaces
- Both interfaces share the same link speed
- SR-IOV and VEA are not supported simultaneously. Only one of the features can be used at any given time.

3.5.2 Hyper-V with VMQ

Mellanox WinOF Rev 4.80.50000 includes a Virtual Machine Queue (VMQ) interface to support Microsoft Hyper-V network performance improvements and security enhancement.

VMQ interface supports:

- Classification of received packets by using the destination MAC address to route the packets to different receive queues
- NIC ability to use DMA to transfer packets directly to a Hyper-V child-partition's shared memory
- Scaling to multiple processors, by processing packets for different virtual machines on different processors.

➢ ***To enable Hyper-V with VMQ using UI:***

Step 1. Open Hyper-V Manager.

Step 2. Right-click the desired Virtual Machine (VM), and left-click Settings in the pop-up menu.

Step 3. In the Settings window, under the relevant network adapter, select “Hardware Acceleration”.

Step 4. Check/uncheck the box “Enable virtual machine queue” to enable/disable VMQ on that specific network adapter.

➢ ***To enable Hyper-V with VMQ using PowerShell:***

Step 1. Enable VMQ on a specific VM: Set-VMNetworkAdapter <VM Name> -VmqWeight 100

Step 2. Disable VMQ on a specific VM: Set-VMNetworkAdapter <VM Name> -VmqWeight 0

3.5.3 Network Virtualization using Generic Routing Encapsulation (NVGRE)

Network Virtualization using Generic Routing Encapsulation (NVGRE) off-load is currently supported in Windows Server 2012 R2 with the latest updates for Microsoft.

Network Virtualization using Generic Routing Encapsulation (NVGRE) is a network virtualization technology that attempts to alleviate the scalability problems associated with large cloud computing deployments. It uses Generic Routing Encapsulation (GRE) to tunnel layer 2 packets across an IP fabric, and uses 24 bits of the GRE key as a logical network discriminator (which is called a tenant network ID).

Configuring the Hyper-V Network Virtualization, requires two types of IP addresses:

- **Provider Addresses (PA)** - unique IP addresses assigned to each Hyper-V host that are routable across the physical network infrastructure. Each Hyper-V host requires at least one PA to be assigned.
- **Customer Addresses (CA)** - unique IP addresses assigned to each Virtual Machine that participate on a virtualized network. Using NVGRE, multiple CAs for VMs running on a Hyper-V host can be tunneled using a single PA on that Hyper-V host. CAs must be unique across all VMs on the same virtual network, but they do not need to be unique across virtual networks with different Virtual Subnet ID.

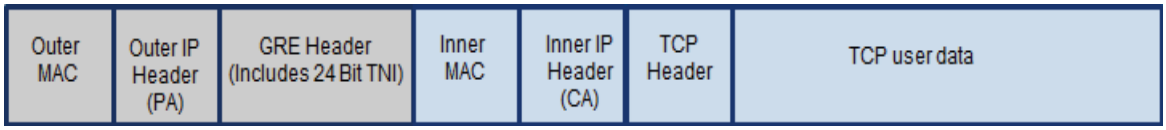
The VM generates a packet with the addresses of the sender and the recipient within the CA space. Then Hyper-V host encapsulates the packet with the addresses of the sender and the recipient in PA space.

PA addresses are determined by using virtualization table. Hyper-V host retrieves the received packet, identifies recipient and forwards the original packet with the CA addresses to the desired VM.

NVGRE can be implemented across an existing physical IP network without requiring changes to physical network switch architecture. Since NVGRE tunnels terminate at each Hyper-V host, the hosts handle all encapsulation and de-encapsulation of the network traffic. Firewalls that block GRE tunnels between sites have to be configured to support forwarding GRE (IP Protocol 47) tunnel traffic.

For further details on configuring NVGRE, please refer to Appendix A,“NVGRE Configuration Scripts Examples,” on page 147

Figure 5: NVGRE Packet Structure



3.5.3.1 Enabling/Disabling NVGRE Offloading

To leverage NVGRE to virtualize heavy network IO workloads, the Mellanox ConnectX®-3 Pro network NIC provides hardware support for GRE off-load within the network NICs by default.

➢ ***To enable/disable NVGRE off-loading:***

Step 1. Open the Device Manager.

Step 2. Go to the Network adapters.

Step 3. Right click 'Properties' on Mellanox ConnectX®-3 Pro Ethernet Adapter card.

Step 4. Go to Advanced tab.

Step 5. Choose the `'Encapsulate Task Offload'` option.

Step 6. Set one of the following values:

- Enable - GRE off-loading is Enabled by default
- Disabled - When disabled the Hyper-V host will still be able to transfer NVGRE traffic, but TCP and inner IP checksums will be calculated by software that significant reduces performance.

3.5.3.1.1 Configuring the NVGRE using PowerShell

Hyper-V Network Virtualization policies can be centrally configured using PowerShell 3.0 and PowerShell Remoting.

Step 1. **[Windows Server 2012 Only]** Enable the Windows Network Virtualization binding on the physical NIC of each Hyper-V Host (Host 1 and Host 2)

```
PS $ Enable-NetAdapterBinding <EthInterfaceName>(a)-ComponentID ms_netwnv
```

<EthInterfaceName> - Physical NIC name

Step 2. Create a vSwitch.

```
PS $ New-VMSwitch <vSwitchName> -NetAdapterName <EthInterfaceName>-AllowManagementOS
$true
```

Step 3. Shut down the VMs.

```
PS $ Stop-VM -Name <VM Name> -Force -Confirm
```

Step 4. Configure the Virtual Subnet ID on the Hyper-V Network Switch Ports for each Virtual Machine on each Hyper-V Host (Host 1 and Host 2).

```
PS $ Add-VMNetworkAdapter -VMName <VMName> -SwitchName <vSwitchName> -StaticMacAddress
<StaticMAC Address>
```

Step 5. Configure a Subnet Locator and Route records on all Hyper-V Hosts (same command on all Hyper-V hosts)

```
PS $ New-NetVirtualizationLookupRecord -CustomerAddress <VMInterfaceIPAddress 1/n> -
ProviderAddress <HypervisorInterfaceIPAddress1> -VirtualSubnetID <virtualsubnetID> -
MACAddress <VMmacaddress1>[a] -Rule "TranslationMethodEncap"

PS $ New-NetVirtualizationLookupRecord -CustomerAddress <VMInterfaceIPAddress 2/n> -
ProviderAddress <HypervisorInterfaceIPAddress2> -VirtualSubnetID <virtualsubnetID> -
MACAddress <VMmacaddress2>[a] -Rule "TranslationMethodEncap"
```

a. This is the VM's MAC address associated with the vSwitch connected to the Mellanox device.

Step 6. Add customer route on all Hyper-V hosts (same command on all Hyper-V hosts).

```
PS $ New-NetVirtualizationCustomerRoute -RoutingDomainID "{11111111-2222-3333-4444-
000000005001}" -VirtualSubnetID <virtualsubnetID> -DestinationPrefix <VMInterfaceIPAd-
dress/Mask> -NextHop "0.0.0.0" -Metric 255
```

Step 7. Configure the Provider Address and Route records on each Hyper-V Host using an appropriate interface name and IP address.

```
PS $ $NIC = Get-NetAdapter <EthInterfaceName>
PS $ New-NetVirtualizationProviderAddress -InterfaceIndex $NIC.InterfaceIndex -Pro-
viderAddress <HypervisorInterfaceIPAddress> -PrefixLength 24
```

```
PS $ New-NetVirtualizationProviderRoute -InterfaceIndex $NIC.InterfaceIndex -Destina-
tionPrefix "0.0.0.0/0" -NextHop <HypervisorInterfaceIPAddress>
```

Step 8. Configure the Virtual Subnet ID on the Hyper-V Network Switch Ports for each Virtual Machine on each Hyper-V Host (Host 1 and Host 2).

```
PS $ Get-VMNetworkAdapter -VMName <VMName> | where {$_.MacAddress -eq <VMmacaddress1>}
| Set-VMNetworkAdapter -VirtualSubnetID <virtualsubnetID>
```

Please repeat steps 5 to 8 on each Hyper-V after rebooting the Hypervisor.

3.5.3.2 Verifying the Encapsulation of the Traffic

Once the configuration using PowerShell is completed, verifying that packets are indeed encapsulated as configured is possible through any packet capturing utility. If configured correctly, an encapsulated packet should appear as a packet consisting of the following headers:

Outer ETH Header, Outer IP, GRE Header, Inner ETH Header, Original Ethernet Payload.

3.5.3.3 Removing NVGRE configuration

Step 1. Set VSID back to 0 (on each Hyper-V for each Virtual Machine where VSID was set)

```
PS $ Get-VMNetworkAdapter <VMName>(a) | where {$_.MacAddress -eq <VMMacAddress>(b)} |
Set-VMNetworkAdapter -VirtualSubnetID 0
```

- VMName - the name of Virtual machine
- VMMacAddress - the MAC address of VM's network interface associated with vSwitch that was connected to Mellanox device.

Step 2. Remove all lookup records (same command on all Hyper-V hosts).

```
PS $ Remove-NetVirtualizationLookupRecord
```

Step 3. Remove customer route (same command on all Hyper-V hosts).

```
PS $ Remove-NetVirtualizationCustomerRoute
```

Step 4. Remove Provider address (same command on all Hyper-V hosts).

```
PS $ Remove-NetVirtualizationProviderAddress
```

Step 5. Remove provider routed for a Hyper-V host.

```
PS $ Remove-NetVirtualizationProviderRoute
```

Step 6. For HyperV running Windows 2012 only disable network adapter binding to ms_netwnv service

```
PS $ Disable-NetAdapterBinding <EthInterfaceName>(a) -ComponentID ms_netwnv
<EthInterfaceName> - Physical NIC name
```

3.5.4 Single Root I/O Virtualization (SR-IOV)

Single Root I/O Virtualization (SR-IOV) is a technology that allows a physical PCIe device to present itself multiple times through the PCIe bus. This technology enables multiple virtual instances of the device with separate resources. Mellanox adapters are capable of exposing in ConnectX®-3/ConnectX®-3 Pro adapter cards, up to 126 virtual instances called Virtual Functions (VFs). These virtual functions can then be provisioned separately. Each VF can be seen as an addition device connected to the Physical Function. It also shares resources with the Physical Function.

SR-IOV is commonly used in conjunction with an SR-IOV enabled hypervisor to provide virtual machines direct hardware access to network resources hence increasing its performance.

This guide demonstrates the setup and configuration of SR-IOV, using Mellanox ConnectX® VPI adapter cards family. SR-IOV VF is a single port device.

Mellanox device is a dual-port single-PCI function. Virtual Functions' pool belongs to both ports. To define how the pool is divided between the two ports use the Powershell `"SriovPort1NumVFs"` command (see Step 5 in Section 3.5.4.4.2, “Enabling SR-IOV in Mellanox WinOF Package (Ethernet SR-IOV Only)”, on page 80).

3.5.4.1 SRIOV Ethernet over Hyper-V

3.5.4.1.1 System Requirements

- A server and BIOS with SR-IOV support. BIOS settings might need to be updated to enable virtualization support and SR-IOV support.
- Hypervisor OS: Windows Server 2012 R2 and above
- Virtual Machine (VM) OS:
  - The VM OS can be either Windows Server 2012 and above
- Mellanox ConnectX®-3/ ConnectX®-3 Pro VPI Adapter Card family with SR-IOV capability
- Mellanox WinOF 4.61 or higher

3.5.4.1.2 Feature Limitations

- SR-IOV is supported only in Ethernet ports and can be enabled if all ports are set as Ethernet.
  - RDMA (i.e RoCE) capability is not available in SR-IOV mode

3.5.4.2 SRIOV InfiniBand over KVM

3.5.4.2.1 System Requirements

- A server and BIOS with SR-IOV support. BIOS settings might need to be updated to enable virtualization support and SR-IOV support.
- Hypervisor OS: Linux KVM using SR-IOV enabled drivers
- Virtual Machine (VM) OS:
  - The VM OS can be Windows Server 2008 R2 and above

    For further details about assigning a VF to the Windows VM, please refer to steps 1-5 in section 4.15.4.1 “Assigning the SR-IOV Virtual Function to the Red Hat KVM VM Server” of the MLNX OFED User Manual.
- Mellanox ConnectX®-3/ ConnectX®-3 Pro VPI Adapter Card family with SR-IOV capability
- Mellanox WinOF 4.80 or higher

3.5.4.2.2 Feature Limitations (Compared to Native InfiniBand)

- The following InfiniBand subnet administration tools are not supported in guest OS:
  - opensm
  - ibdump
  - ibutils
  - ibutils2
  - infininband-diags
- For a UD QP, only SGID index 0 is supported.
- The allocation of the GIDs (per port) in the VFs are accordingly:
  - 16 GIDs are allocated to the PF
  - 2 GIDs are allocated to every VF
  - The remaining GIDs (if such exist), will be assigned to the VFs, one GID to every VF - starting from the lower VF.
- Currently, Mellanox IB Adapter Diagnostic Counters and Mellanox IB Adapter Traffic Counters are not supported.
- Only Administrator assigned GUIDs are supported, please refer to the MLNX_OFED User Manual for instructions on how to configure Administrator assigned GUIDs.

3.5.4.3 Configuring SR-IOV Host Machines

The following are the necessary steps for configuring host machines:

3.5.4.3.1 Enabling SR-IOV in BIOS

Depending on your system, perform the steps below to set up your BIOS. The figures used in this section are for illustration purposes only.

For further information, please refer to the appropriate BIOS User Manual.

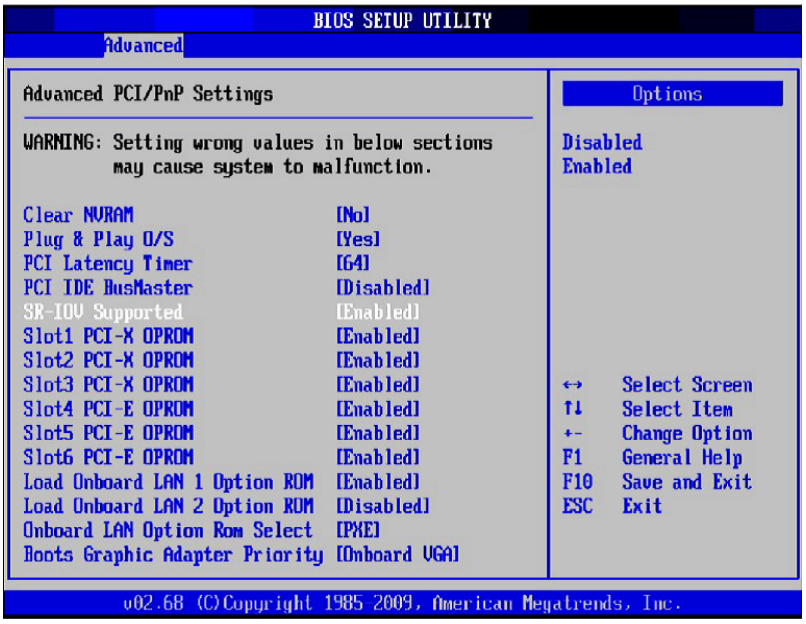
➢ ***To enable SR-IOV in BIOS:***

Step 1. Make sure the machine’s BIOS supports SR-IOV.

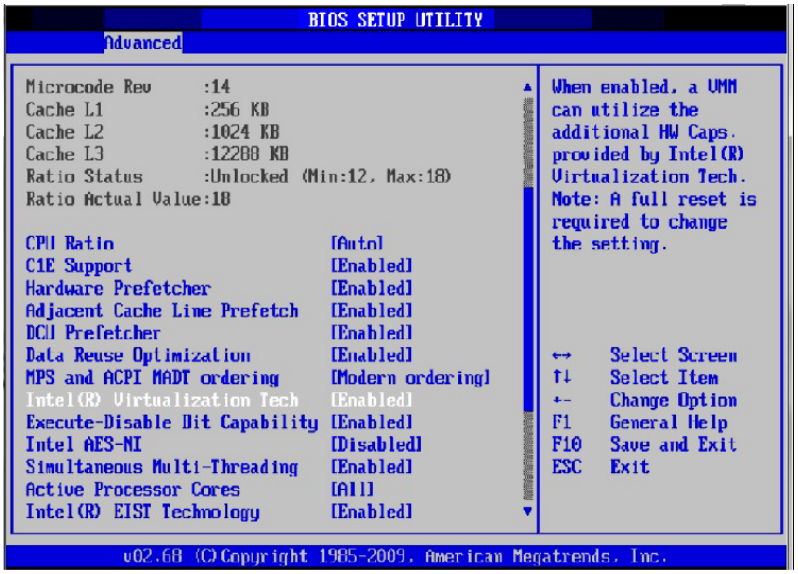
Please, consult BIOS vendor website for SR-IOV supported BIOS versions list. Update the BIOS version if necessary.

Step 2. Follow BIOS vendor guidelines to enable SR-IOV according to BIOS User Manual. For example,

a. Enable SR-IOV.



b. Enable “Intel Virtualization Technology” Support



For further details, please refer to the vendor's website.

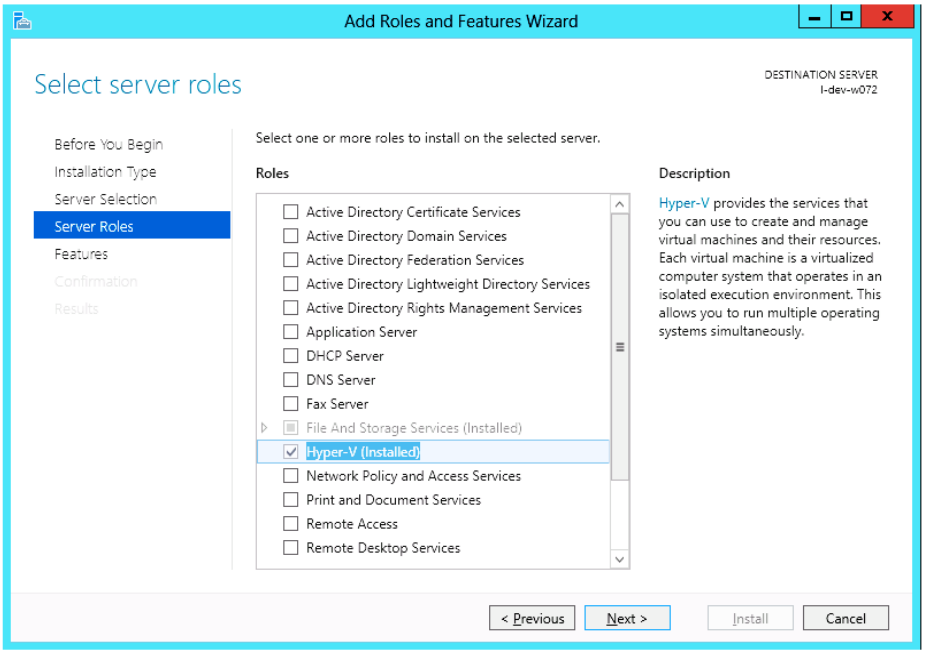
3.5.4.3.2 Installing Hypervisor Operating System (SR-IOV Ethernet Only)

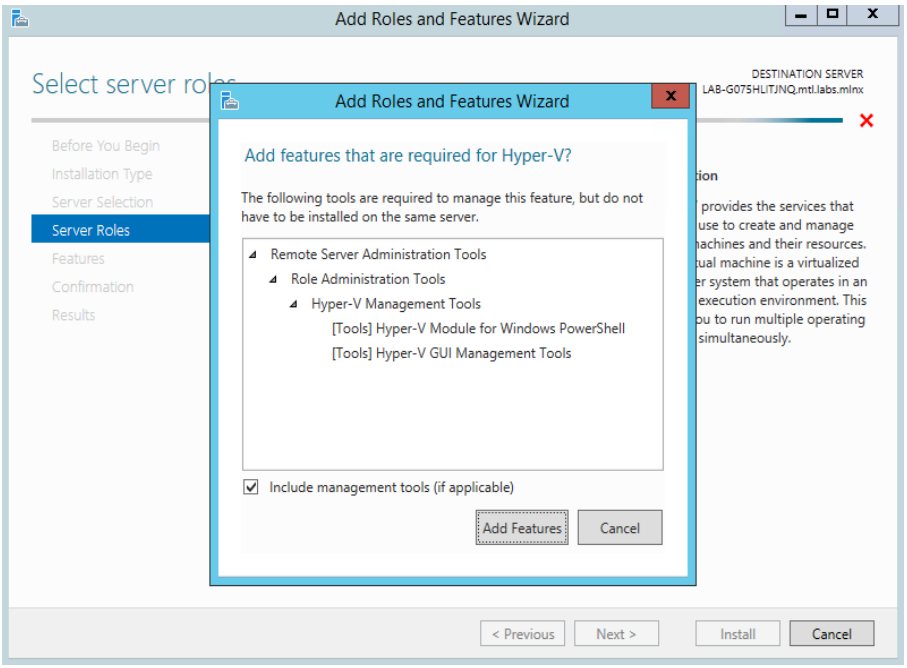
➢ ***To install Hypervisor Operating System:***

Step 1. Install Windows Server 2012 R2 and above.

Step 2. Install Hyper-V role:

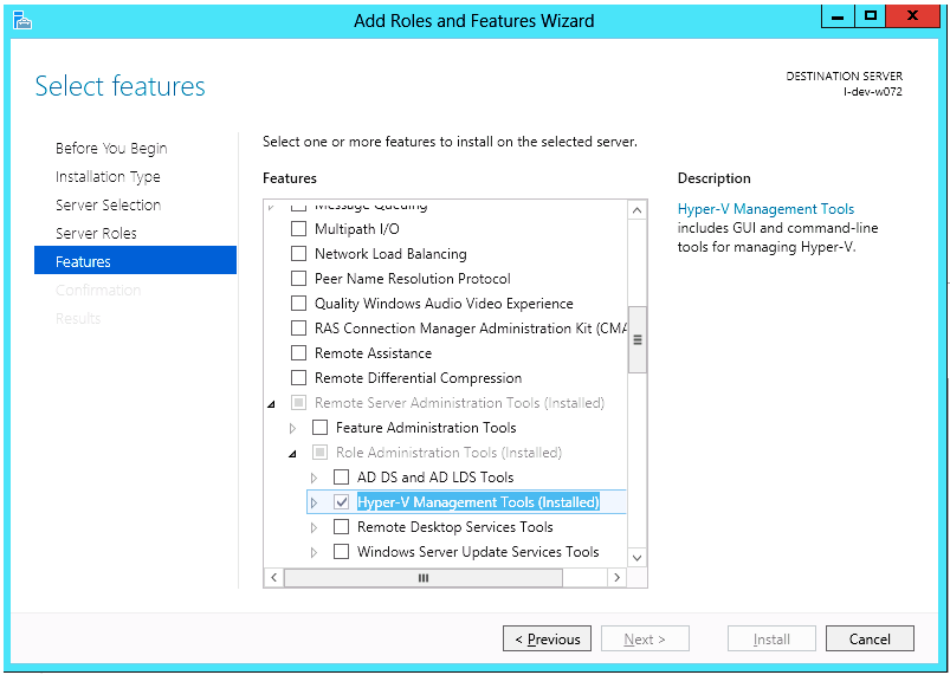
- Go to: Server Manager -> Manage -> Add Roles and Features and set the following:
  - Installation Type -> Role-based or Feature-based Installation
  - Server Selection -> Select a server fro the server pool
  - Server Roles -> Hyper-V (see figures below)



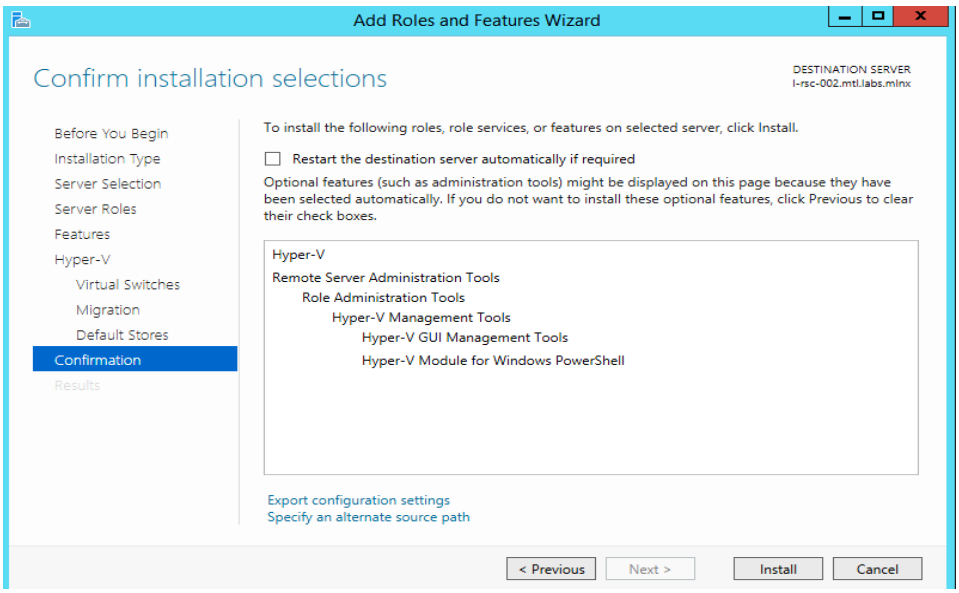


Step 3. Install Hyper-V Management Tools.

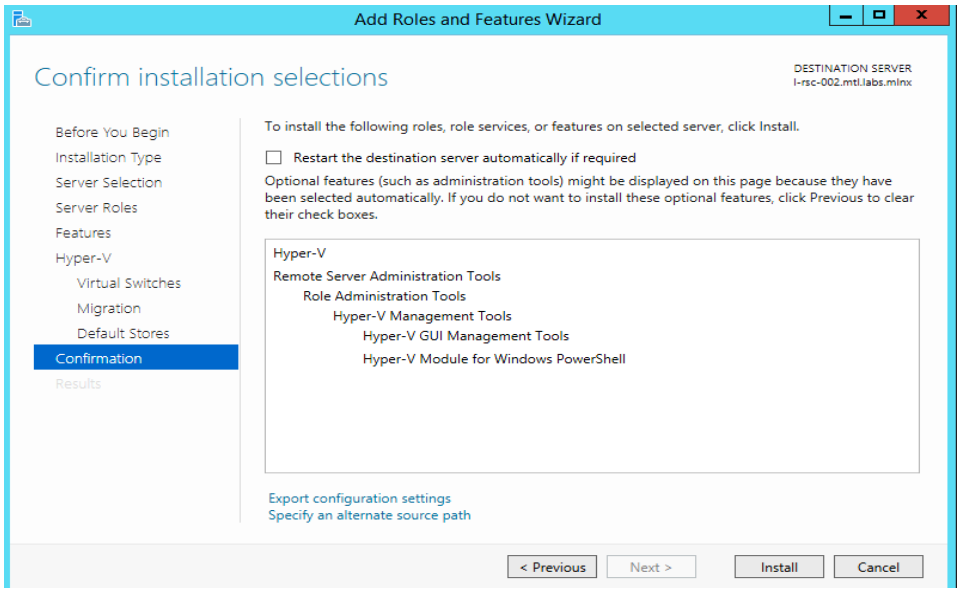
Features - > Remote Server Administration Tools -> Role Administration Tools -> Hyper-V Administration Tool



Step 4. Confirm the Installation.



Step 5. Click Install.



Step 6. Reboot the system.

3.5.4.3.3 Verifying SR-IOV Support within the Host Operating System (SR-IOV Ethernet Only)

➢ ***To verify that the system is properly configured for SR-IOV:***

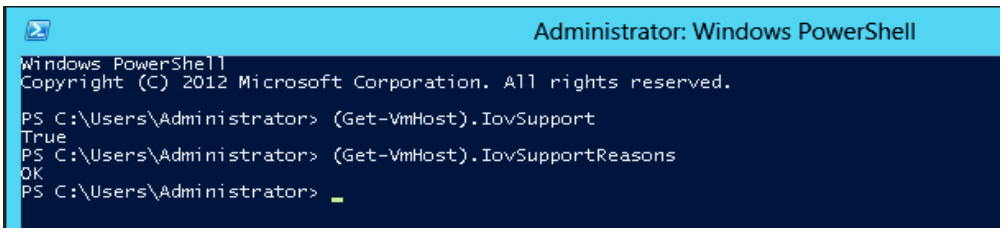
Step 1. Go to: Start-> Windows Powershell.

Step 2. Run the following PowerShell commands.

```
PS $ (Get-VmHost).IovSupport
PS $ (Get-VmHost).IovSupportReasons
```

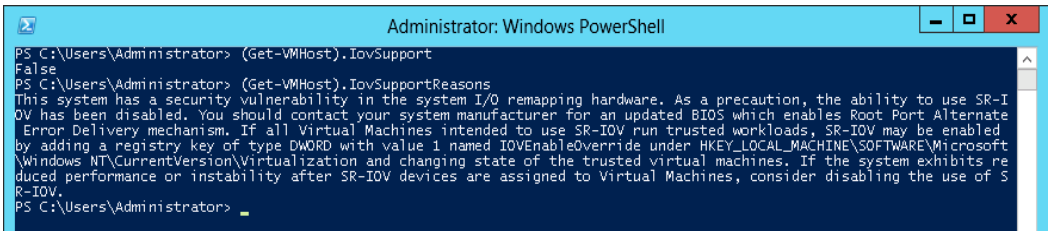
In case that SR-IOV is supported by the OS, the output in the PowerShell is as in the figure below.

Figure 6: Operating System Supports SR-IOV



Note: If BIOS was updated according to BIOS vendor instructions and you see the message displayed in the figure below, update the registry configuration as described in the (Get-VmHost).IovSupportReasons message.

Figure 7: SR-IOV Support



Step 3. Reboot

Step 4. Verify the system is configured correctly for SR-IOV as described in Steps 1/2.

3.5.4.3.4 Creating a Virtual Machine (SR-IOV Ethernet Only)

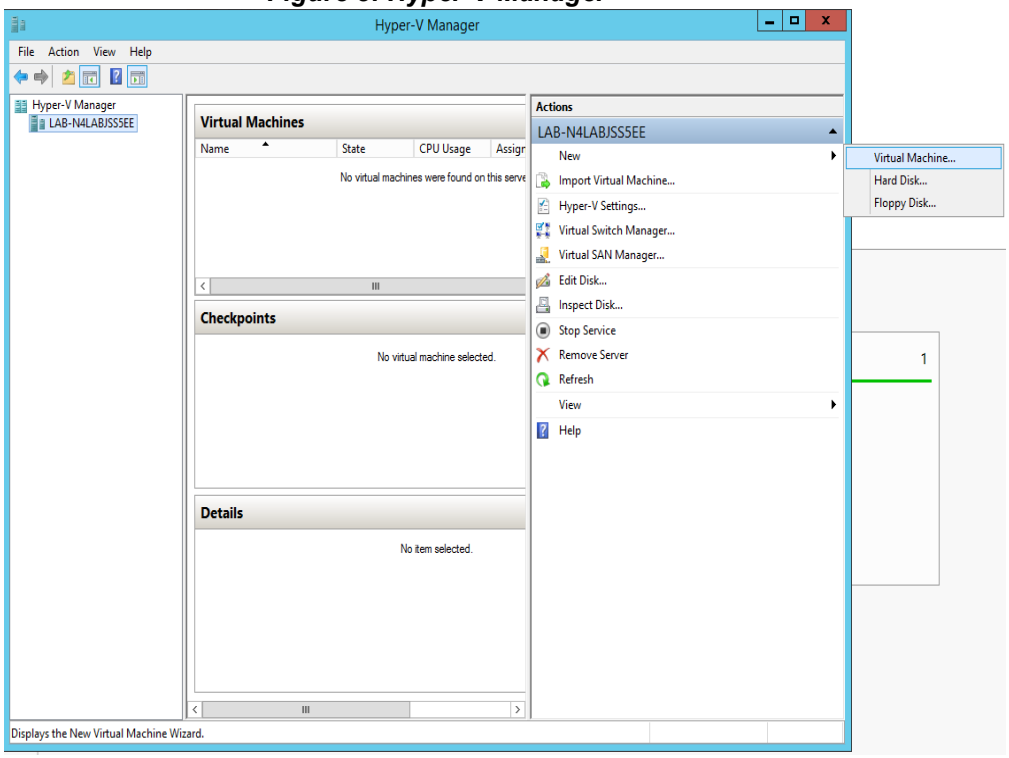
➢ ***To create a virtual machine***

Step 1. Go to: Server Manager -> Tools -> Hyper-V Manager.

Step 2. Go to: New->Virtual Machine and set the following:

- Name: <name>
- Startup memory: 4096 MB
- Connection: Not Connected

Figure 8: Hyper-V Manager

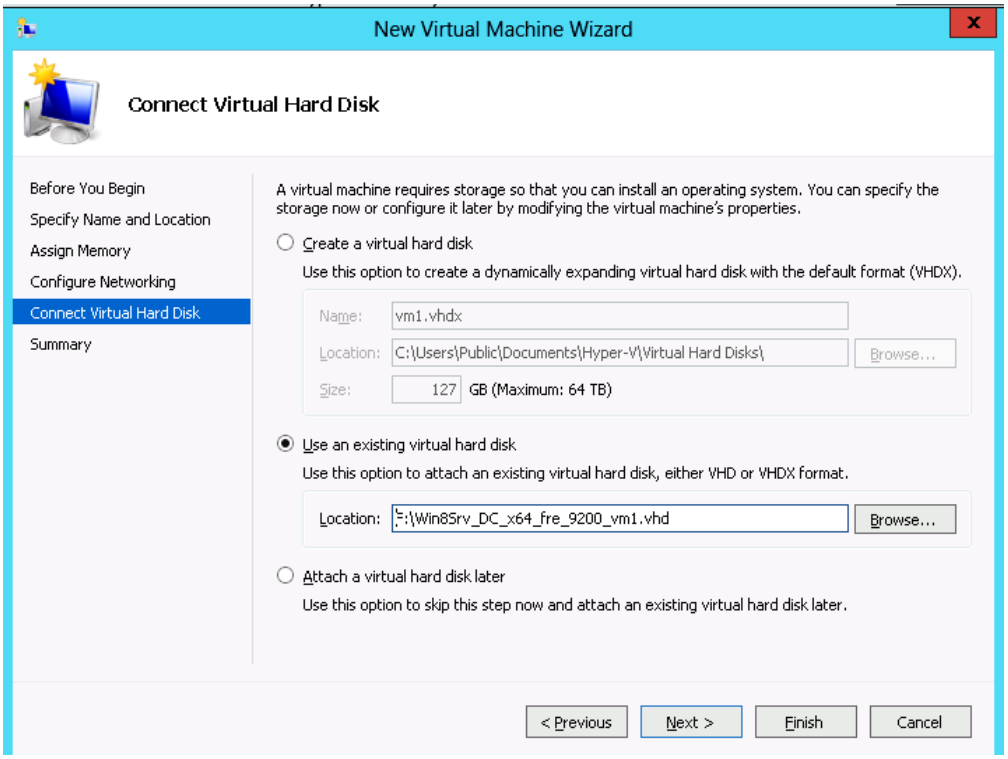


Step 3. Connect the virtual hard disk in the New Virtual Machine Wizard.

Step 4. Go to: Connect Virtual Hard Disk -> Use an existing virtual hard disk.

Step 5. Select the location of the vhd file.

Figure 9: Connect Virtual Hard Disk



3.5.4.4 Configuring Mellanox Network Adapter for SR-IOV

The following are the steps for configuring Mellanox Network Adapter for SR-IOV:

3.5.4.4.1 Enabling SR-IOV in Firmware

For non-Mellanox (OEM branded cards) you may need to download and install new firmware. For the latest OEM firmware, please go to:
http://www.mellanox.com/page/oem_firmware_download

As of firmware version 2.31.5000, SR-IOV can be enabled and managed by using the mlxconfig too. For older firmware versions, use the flint tool.

➢ ***To enable SR-IOV using mlxconfig:***

mlxconfig is part of MFT tools used to simplify firmware configuration. The tool is available with MFT tools 3.6.0 or higher.

Step 1. Download MFT for Windows.

www.mellanox.com > Products > Software > Firmware Tools

Step 2. Get the device ID (look for the "_pciconf" string in the output).

```
> mst status
```

Example:

```
MST devices:
------------
  mt4103_pci_cr0
  mt4103_pciconf0
```

Step 3. Check the current SR-IOV configuration.

```
> mlxconfig -d mt4103_pciconf0 q
```

Example:

```
Device #1:
----------

Device type:    ConnectX3Pro
PCI device:     mt4103_pciconf0

Configurations:          Current
         SRIOV_EN        N/A
         NUM_OF_VFS      N/A
         WOL_MAGIC_EN_P2 N/A
         LINK_TYPE_P1    N/A
         LINK_TYPE_P2    N/A
```

Step 4. Enable SR-IOV with 16 VFs.

```
> mlxconfig -d mt4103_pciconf0 s SRIOV_EN=1 NUM_OF_VFS=16
```

Warning: Care should be taken in increasing the number of VFs. All servers are guaranteed to support 16 VFs. More VFs can lead to exceeding the BIOS limit of MMIO available address space.

Example:

```
Device #1:
----------

Device type:    ConnectX3Pro
PCI device:     mt4103_pciconf0

Configurations:          Current New
         SRIOV_EN        N/A     1
         NUM_OF_VFS      N/A     16
         WOL_MAGIC_EN_P2 N/A     N/A
         LINK_TYPE_P1    N/A     N/A
         LINK_TYPE_P2    N/A     N/A

 Apply new Configuration? ? (y/n) [n] : y
Applying... Done!
-I- Please reboot machine to load new configurations.
```

Step 5. Reboot the machine (After the reboot, continue to Section 3.5.4.4.2, “Enabling SR-IOV in Mellanox WinOF Package (Ethernet SR-IOV Only)”, on page 80).

➢ ***To enable SR-IOV using flint:***

Step 1. Download MFT for Windows.

www.mellanox.com > Products > Software > Firmware Tools

Step 2. Get the device ID (look for the "_pciconf" string in the output).

```
> mst status
```

Example:

```
MST devices:
------------
  mt4103_pci_cr0
  mt4103_pciconf0
```

Step 3. Verify that HCA is configured for SR-IOV by dumping the device configuration file to user-chosen location <ini device file>.ini.

```
> flint -d <device> dc > <ini device file>.ini
```

Step 4. Verify in the [HCA] section of the .ini that the following fields appear:

```
[HCA]
num_pfs = 1
total_vfs = 16
sriov_en = true
```

Warning: Care should be taken in increasing the number of VFs. All servers are guaranteed to support 16 VFs. More VFs can lead to exceeding the BIOS limit of MMIO available address space.

Step 5. If the fields do not appear, please, edit the .ini file and add them manually.

Parameter	Recommended Value
num_pfs	1 **Note**: This field is optional and might not always appear.
total_vfs	<0-126> (The chosen value should be within BIOS limit of MMIO available address space)
sriov_en	true

Step 6. Create a binary image using the modified ini file.

```
> mlxburn -fw <fw name>.mlx -conf <ini device file>.ini -wrimage <file
name>.bin
```

Step 7. Burn the firmware.

The file <file name>.bin is a firmware binary file with SR-IOV enabled that has 16 VFs.

```
> flint -dev <PCI device> -image <file name>.bin b
```

Step 8. Reboot the system for changes to take effect.

For more information, please, contact Mellanox Support.

3.5.4.4.2 Enabling SR-IOV in Mellanox WinOF Package (Ethernet SR-IOV Only)

➢ ***To enable SR-IOV in Mellanox WinOF Package***

Step 1. Install Mellanox WinOF package that supports SR-IOV.

Step 2. Configure HCA ports' type to Ethernet.
For further information, please refer to Section 3.1.1, “Port Configuration”, on page 29.

Note: SR-IOV cannot be enabled if one of the ports is InfiniBand.

Step 3. Set the Execution Policy.

```
PS $ Set-ExecutionPolicy AllSigned

Execution Policy Change
The execution policy helps protect you from scripts that you do not trust. Changing the
execution policy might expose
you to the security risks described in the about_Execution_Policies help topic at
http://go.microsoft.com/fwlink/?LinkID=135170. Do you want to change the execution pol-
icy?
[Y] Yes  [N] No  [S] Suspend  [?] Help (default is "Y"): y
```

Step 4. Query SR-IOV configuration with Powershell.

```
PS $ Get-MlnxPCIDeviceSriovSetting
```

Example:

```
Caption         : MLNX_PCIDeviceSriovSettingData 'Mellanox ConnectX-3 PRO VPI (MT04103)
Network Adapter'
Description     : Mellanox ConnectX-3 PRO VPI (MT04103) Network Adapter
ElementName     : HCA 0
InstanceID      : PCI\VEN_15B3&DEV_1007&SUBSYS_22F5103C&REV_00\24BE05FFFFB9E2E000
Name            : HCA 0
Source          : 3
SystemName      : LAB-N4LABJSS5EE
SriovEnable     : False
SriovPort1NumVFs : 16
SriovPort2NumVFs : 0
SriovPortMode   : 0
PSComputerName  :
```

Step 5. Enable SR-IOV through Powershell on both ports.[1]

```
PS $ Set-MlnxPCIDeviceSriovSetting -Name "HCA 0" -SriovEnable $true -SriovPortMode 2
-SriovPort1NumVFs 8 -SriovPort2NumVFs 8
```

Example:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "SetValue" on target "MLNX_PCIDeviceSriovSettingData:
MLNX_PCIDeviceSriovSettingData 'Mellanox
 ConnectX-3 PRO VPI (MT04103) Network Adapter' (InstanceID =
"PCI\VEN_15B3&DEV_1007&SUBSYS_22F5103C&R...)".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is "Y"):
y
```

Mellanox device is a dual-port single-PCI function. Virtual Functions' pool belongs to both ports. To define how the pool is divided between the two ports use the Powershell "SriovPort1NumVFs" command.

1. **SriovPortMode 2** - Enables SR-IOV on both ports.
   SriovPort1NumVFs 8 & **SriovPort2NumVFs 8** - Enable 8 Virtual Functions for each port when working in manual mode. By default, there are assigned 16 virtual functions on the first port.

SR-IOV mode configuration parameters:

Parameter Name	Values	Description
SriovEnable	• 0 = RoCE (default) • 1 = SR-IOV	Configures the RDMA or SR-IOV mode. The default WinOF configuration mode is RoCE. To switch to SR-IOV, set the `SriovEnable` registry key value to 1. By default in SR-IOV mode, all VF pool belongs to Port 1. To change the VF pool distribution, change the `PortMode` to manual and choose how many VFs to assign to each port. **Note**: RDMA is not supported in SR-IOV mode.
SriovPortMode	• 0 = auto_port1 (default) • 1 = auto_port2 • 2 = manual	Configures the number of VFs to be enabled by the bus driver to each port. **Note**: In auto_portX mode, port X will have the number of VFs according to the burnt value in the device and the other port will have no SR-IOV and it will support native Ethernet (i.e. no RoCE). Setting this parameter to "Manual" will configure the number of VFs for each port according to the registry key MaxVFPortX. **Note**: The number of VFs can be configured both on a Mellanox bus driver level and Network Interface level (i.e using Set-NetAdapterSriov Powershell cmdlet). The number of VFs actually available to the Network Interface is the minimum value between mellanox bus driver configuration and Network Interface configuration. For example, if 8 VFs support was burnt in firmware, SriovPortMode is auto_port1, and Network Interface was allowed 32 VFs using SetNetAdapterSriov Powershell cmdlet, the actual number of VFs available to Network Interface will be 8.

Parameter Name	Values	Description
SriovPort1NumVFs SriovPort2NumVFs	• 16=(default)	SriovPort<i>NumVFs The maximum number of VFs that are allowed per port. This is the number of VFs the bus driver will open when working in manual mode. **Note**: If the total number of VFs requested is larger than the number of VFs burnt in firmware, each port X(1\2) will have the number of VFs according to the following formula: (SriovPortXNumVFs / (SriovPort1NumVFs+SriovPort2NumVFs)) *number of VFs burnt in firmware.

Step 6. Verify the new values were set correctly.

```
PS $ Get-MlnxPCIDeviceSriovSetting
```

Example:

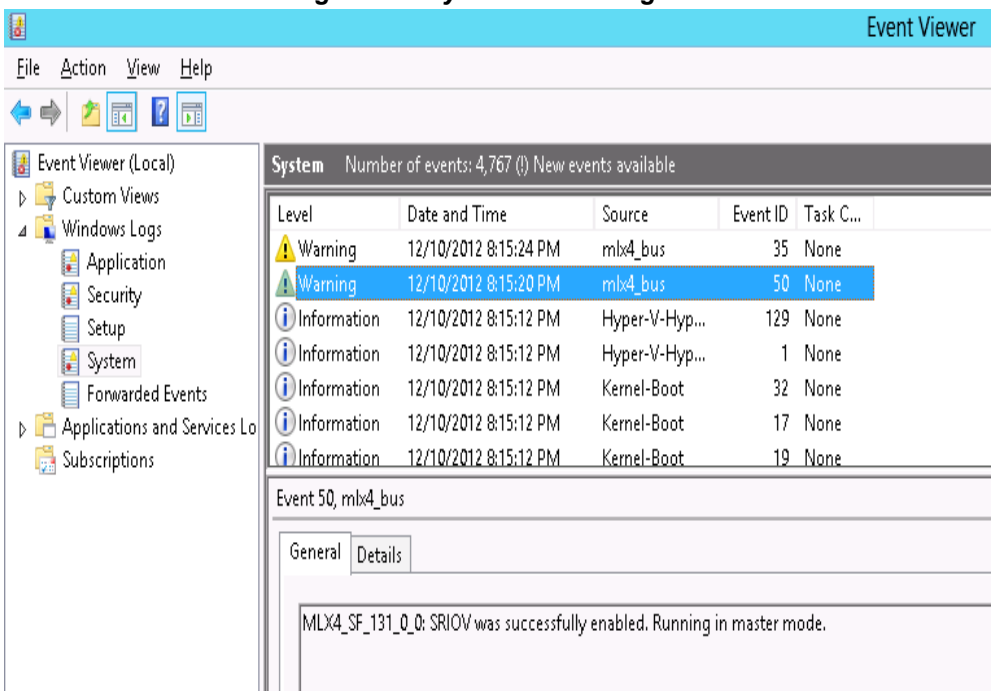
```
Caption         : MLNX_PCIDeviceSriovSettingData 'Mellanox ConnectX-3 PRO VPI (MT04103)
Network Adapter'
Description     : Mellanox ConnectX-3 PRO VPI (MT04103) Network Adapter
ElementName     : HCA 0
InstanceID      : PCI\VEN_15B3&DEV_1007&SUBSYS_22F5103C&REV_00\24BE05FFFFB9E2E000
Name            : HCA 0
Source          : 3
SystemName      : LAB-N4LABJSS5EE
SriovEnable     : True
SriovPort1NumVFs : 8
SriovPort2NumVFs : 8
SriovPortMode   : 2
PSComputerName  :
```

Step 7. Check in the System Event Log that SR-IOV is enabled.

Step a. Open the View Event Logs/Event Viewer.
Go to: Start -> Control Panel -> System and Security -> Administrative Tools -> View Event Logs/Event Viewer

Step b. Open the System logs.
Event Viewer (Local) -> Windows Logs -> System

Figure 10: System Event Log



3.5.4.5 Configuring Operating Systems

3.5.4.5.1 Configuring Virtual Machine Networking (InfiniBand SR-IOV Only)

For further details on enabling/configuring SR-IOV on KVM, please refer to section 3.5.1 Single Root IO Virtualization (SR-IOV) in MLNX OFED User Manual.

3.5.4.5.2 Configuring Virtual Machine Networking (Ethernet SR-IOV Only)

➢ ***To configure Virtual Machine networking:***

Step 1. Create an SR-IOV-enabled Virtual Switch over Mellanox Ethernet Adapter.

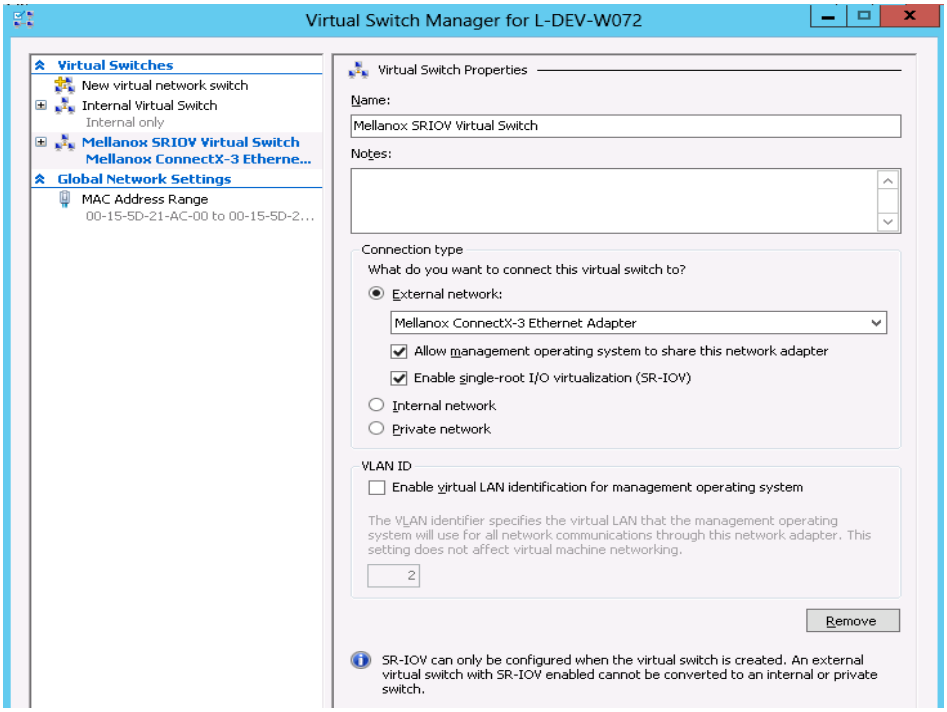
Go to: Start -> Server Manager -> Tools -> Hyper-V Manager

In the Hyper-V Manager: Actions -> Virtual SwitchManager -> External-> Create Virtual Switch

Step 2. Set the following:

- Name:
- External network:
- Enable single-roo I/O virtualization (SR-IOV)

Figure 11: Virtual Switch with SR-IOV



Step 3. Click **Apply**.

Step 4. Click **OK**.

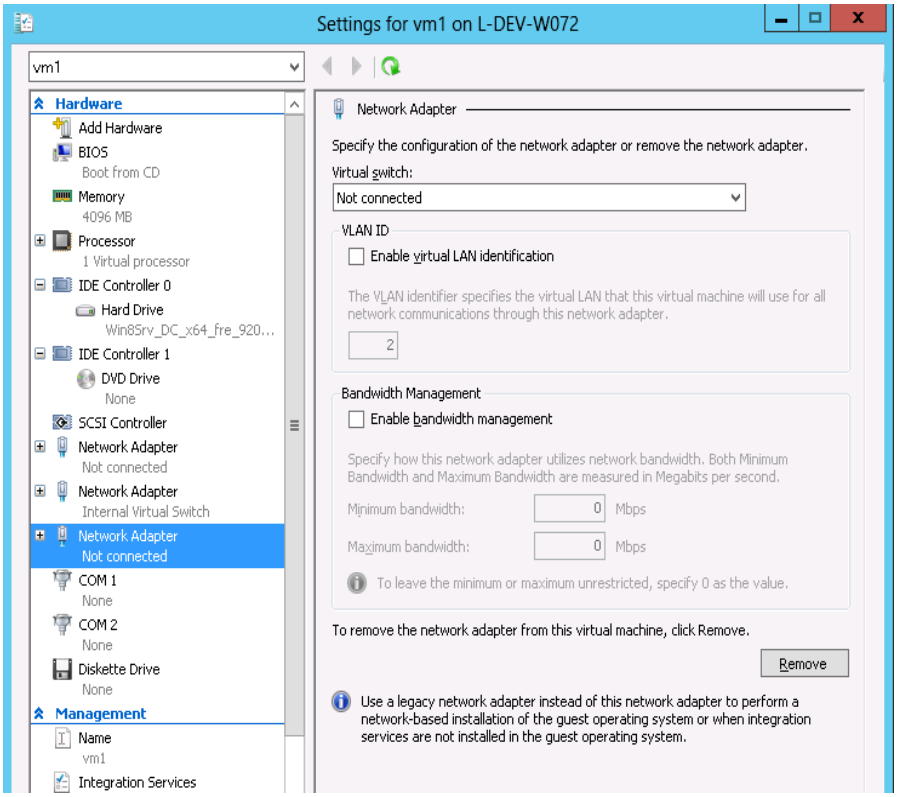
Step 5. Add a VMNIC connected to a Mellanox vSwitch in the VM hardware settings.

In the Hyper-V Manager, right click the VM and go to:

Settings -> Add New Hardware-> Network Adapter-> OK.

In “Virtual Switch” dropdown box, choose Mellanox SR-IOV Virtual Switch.

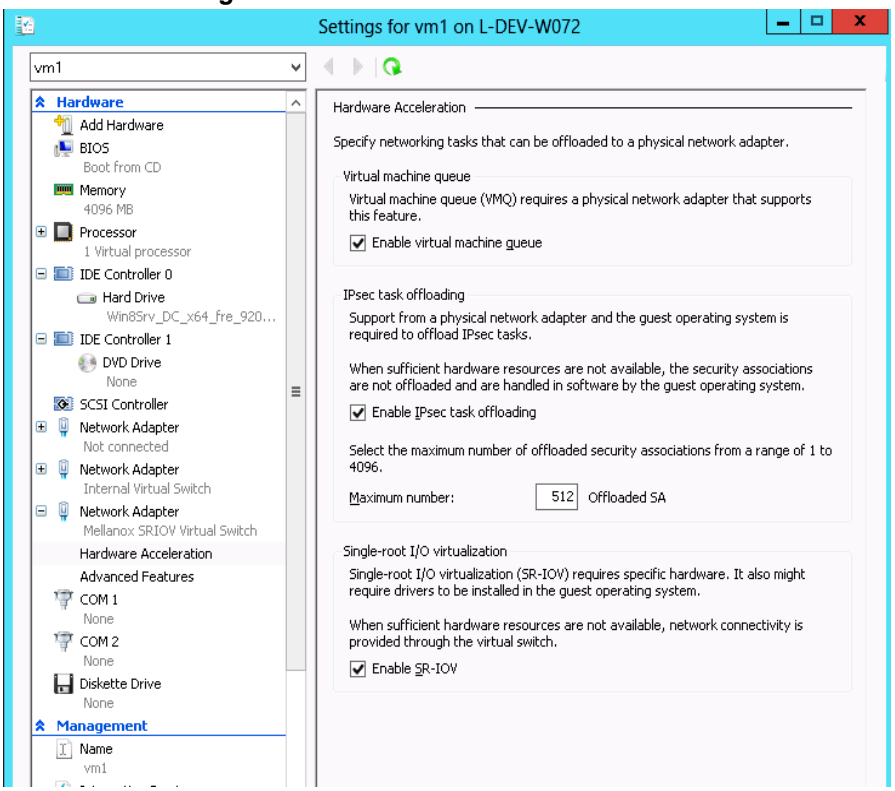
Figure 12: Adding a VMNIC to a Mellanox V-switch



Step 6. Enable the SR-IOV for Mellanox VMNIC.

1. Open VM settings Wizard.
2. Open the Network Adapter and choose Hardware Acceleration.
3. Tick the “Enable SR-IOV” option.
4. Click OK.

Figure 13: Enable SR-IOV on VMNIC



Step 7. Start and connect to the Virtual Machine:

Select the newly created Virtual Machine and go to: Actions panel-> Connect.
In the virtual machine window go to: Actions-> Start

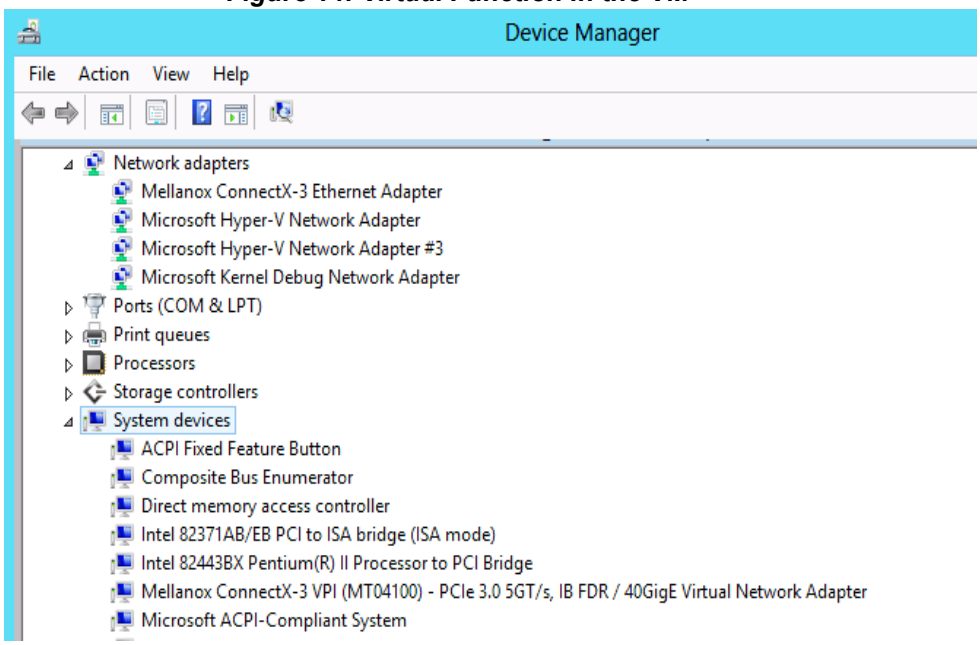
Step 8. Copy the WinOF driver package to the VM using Mellanox VMNIC IP address.

Step 9. Install WinOF driver package on the VM.

Step 10. Reboot the VM at the end of installation.

Step 11. Verify that Mellanox Virtual Function appears in the device manager.

Figure 14: Virtual Function in the VM



To achieve best performance on SR-IOV VF, please run the following powershell commands on the host:

For 10Gbe:

PS $ Set-VMNetworkAdapter -Name "Network Adapter" -VMName vm1 -IovQueuePairsRequested 4

For 40Gbe and 56Gbe:

PS $ Set-VMNetworkAdapter -Name "Network Adapter" -VMName vm1 -IovQueuePairsRequested 8

3.6 Configuration Using Registry Keys

Mellanox IPoIB and Ethernet drivers use registry keys to control the NIC operations. The registry keys receive default values during the installation of the Mellanox adapters. Most of the parameters are visible in the registry by default, however, certain parameters must be created in order to modify the default behavior of the Mellanox driver.

The adapter can be configured either from the User Interface (Device Manager -> Mellanox Adapter -> Right click -> Properties) or by setting the registry directly.

All Mellanox adapter parameters are located in the registry under the following registry key:

```
HKEY_LOCAL_MACHINE
\SYSTEM
\CurrentControlSet
\ Control
\ Class
\{4D36E972-E325-11CE-BFC1-08002bE10318}
\<Index>
```

The registry key can be divided into 4 different groups:

Group	Description
Basic	Contains the basic configuration.
Offload Options	Controls the offloading operation that the NIC supports.
Performance Options	Controls the NIC operation in different environments and scenarios.
Flow Control Options	Controls the TCP/IP traffic.

Any registry key that starts with an asterisk ("*") is a well-known registry key. For more details regarding the registries, please refer to:

http://msdn.microsoft.com/en-us/library/ff570865(v=VS.85).aspx

3.6.1 Finding the Index Value of the HCA

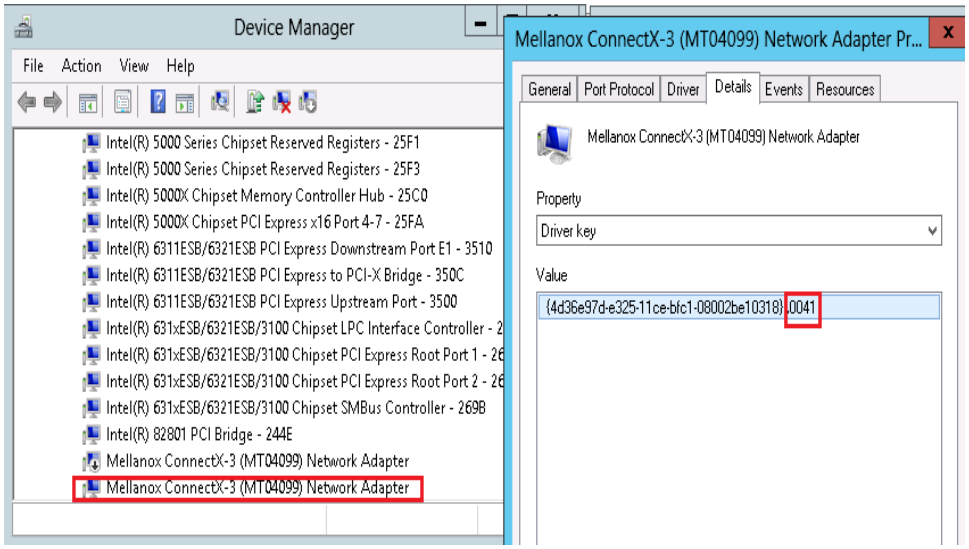
➢ ***To find the nn value of your HCA from the Device Manager please perform the following steps:***

Step 1. Open Device Manager, and go to System devices.

Step 2. Right click -> properties on Mellanox -ConnectX® card.

Step 3. Go to Details tab.

Step 4. Select the Driver key, and obtain the nn number.
In the below example, the index equals 0041



3.6.2 Finding the Index Value of the Network Interface

To find the index value of your Network Interface from the Device Manager please perform the following steps:

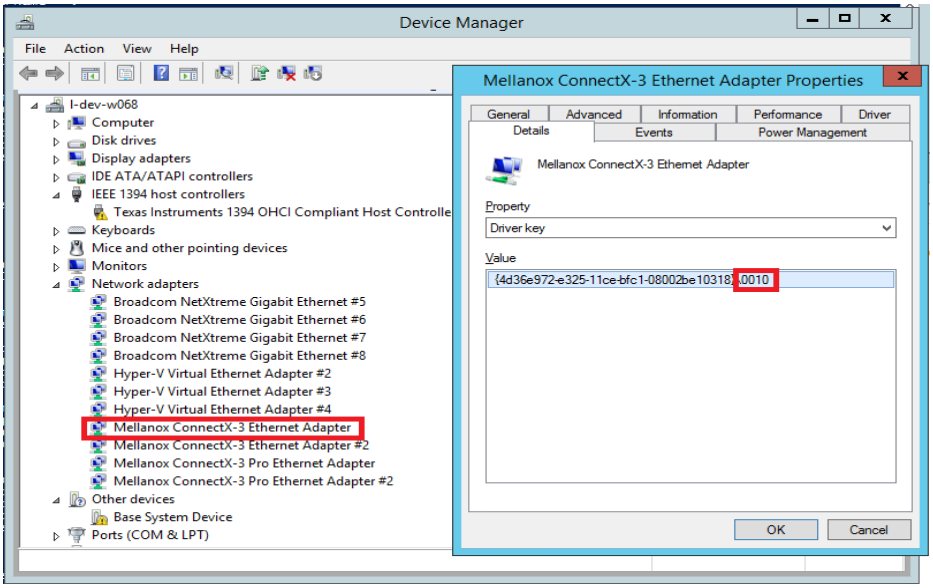
Step 1. Open Device Manager, and go to Network Adapters.

Step 2. Right click ->Properties on Mellanox Connect-X® Ethernet Adapter.

Step 3. Go to Details tab.

Step 4. Select the Driver key, and obtain the nn number.

In the below example, the index equals 0010



3.6.3 Basic Registry Keys

This group contains the registry keys that control the basic operations of the NIC

Value Name	Default Value	Description
*JumboPacket	1500	The maximum size of a frame (or a packet) that can be sent over the wire. This is also known as the maximum transmission unit (MTU). The MTU may have a significant impact on the network's performance as a large packet can cause high latency. However, it can also reduce the CPU utilization and improve the wire efficiency. The standard Ethernet frame size is 1514 bytes, but Mellanox drivers support wide range of packet sizes. The valid values are: • Ethernet: 600 up to 9600 • IPoIB: 1500 up to 4092 **Note**: All the devices across the network (switches and routers) should support the same frame size. Be aware that different network devices calculate the frame size differently. Some devices include the header, i.e. information in the frame size, while others do not. Mellanox adapters do not include Ethernet header information in the frame size. (i.e when setting *JumboPacket to 1500, the actual frame size is 1514).

Value Name	Default Value	Description
*ReceiveBuffers	1024	The number of packets each ring receives. This parameter affects the memory consumption and the performance. Increasing this value can enhance receive performance, but also consumes more system memory. In case of lack of received buffers (dropped packets or out of order received packets), you can increase the number of received buffers. The valid values are 256 up to 4096. **Note**: On 32-bit systems, the non-pageable memory is limited. As a result, when the MTU is higher than 5000 and the ring size is 2048 or more, the initialization can fail due to a lack of memory. If the MTU is more than 5000, the driver limits the ring size on 32-bit system to be 1024.
*TransmitBuffers	2048	The number of packets each ring sends. Increasing this value can enhance transmission performance, but also consumes system memory. The valid values are 256 up to 4096.
*SpeedDuplex	7 (default)	The Speed and Duplex settings that a device supports. This registry key should not be changed and it can be used to query the device capability. Mellanox ConnectX device is set to 7 meaning10Gbps and Full Duplex. **Note**: Default value should not be modified.
MaxNumOfMCList	128	The number of multicast addresses that are filtered by the NIC. If the OS uses more multicast addresses than were defined, it sets the port to multicast promiscuous and the multicast addresses are filtered by OS at protocol level. The valid values are 64 up to 1024. **Note**: This registry value is not exposed via the UI.
*QOS	1	Enables the NDIS Quality of Service (QoS) The valid values are: • 1: enable • 0: disable **Note**: This keyword is only valid for ConnectX-3 when using Windows Server 2012 and above.

Value Name	Default Value	Description
RxIntModerationProfile	1	Enables the assignment of different interrupt moderation profiles for receive completions. Interrupt moderation can have a great effect on optimizing network throughput and CPU utilization. The valid values are: • 0: Low Latency Implies higher rate of interrupts to achieve better latency, or to handle scenarios where only a small number of streams are used. • 1: Moderate Interrupt moderation is set to midrange defaults to allow maximum throughput at minimum CPU utilization for common scenarios. • 2: Aggressive Interrupt moderation is set to maximal values to allow maximum throughput at minimum CPU utilization, for more intensive, multi-stream scenarios.
TxIntModerationProfile	1	Enables the assignment of different interrupt moderation profiles for send completions. Interrupt moderation can have great effect on optimizing network throughput and CPU utilization. The valid values are: • 0: Low Latency Implies higher rate of interrupts to achieve better latency, or to handle scenarios where only a small number of streams are used. • 1: Moderate Interrupt moderation is set to midrange defaults to allow maximum throughput at minimum CPU utilization for common scenarios. • 2: Aggressive Interrupt moderation is set to maximal values to allow maximum throughput at minimum CPU utilization for more intensive, multi-stream scenarios.

3.6.4 Off-load Registry Keys

This group of registry keys allows the administrator to specify which TCP/IP offload settings are handled by the adapter rather than by the operating system.

Enabling offloading services increases transmission performance. Due to offload tasks (such as checksum calculations) performed by adapter hardware rather than by the operating system (and, therefore, with lower latency). In addition, CPU resources become more available for other tasks.

Value Name	Default Value	Description
*LsoV1IPv4	1	Large Send Offload Version 1 (IPv4). The valid values are: • 0: disable • 1: enable
*LsoV1IPv4	1	Large Send Offload Version 2 (IPv4). The valid values are: • 0: disable • 1: enable
*LsoV1IPv4	1	Large Send Offload Version 2 (IPv6). The valid values are: • 0: disable • 1: enable
LSOSize	32000	The maximum number of bytes that the TCP/IP stack can pass to an adapter in a single packet. This value affects the memory consumption and the NIC performance. The valid values are MTU+1024 up to 64000. **Note:** This registry key is not exposed to the user via the UI. If LSOSize is smaller than MTU+1024, LSO will be disabled.
LSOMinSegment	2	The minimum number of segments that a large TCP packet must be divisible by, before the transport can offload it to a NIC for segmentation. The valid values are 2 up to 32. **Note**: This registry key is not exposed to the user via the UI.
LSOTcpOptions	1	Enables that the miniport driver to segment a large TCP packet whose TCP header contains TCP options. The valid values are: • 0: disable • 1: enable **Note**: This registry key is not exposed to the user via the UI.

Value Name	Default Value	Description
LSOIpOptions	1	Enables its NIC to segment a large TCP packet whose IP header contains IP options. The valid values are: • 0: disable • 1: enable **Note**: This registry key is not exposed to the user via the UI.
*IPChecksumOffloadIPv4	3	Specifies whether the device performs the calculation of IPv4 checksums. The valid values are: • 0: (disable) • 1: (Tx Enable) • 2: (Rx Enable) • 3: (Tx and Rx enable)
*TCPUDPChecksumOffloadIPv4	3	Specifies whether the device performs the calculation of TCP or UDP checksum over IPv4. The valid values are: • 0: (disable) • 1: (Tx Enable) • 2: (Rx Enable) • 3: (Tx and Rx enable)
*TCPUDPChecksumOffloadIPv6	3	Specifies whether the device performs the calculation of TCP or UDP checksum over IPv6. The valid values are: • 0: (disable) • 1: (Tx Enable) • 2: (Rx Enable) • 3: (Tx and Rx enable)
ParentBusRegPath	HKLM\SYSTEM\CurrentControlSet\Control\Class\{4d36e97d-e325-11ce-bfc1-08002be10318}\0073	TCP checksum off-load IP-IP.

3.6.5 Performance Registry Keys

This group of registry keys configures parameters that can improve adapter performance.

Value Name	Default Value	Description
RecvCompletion-Method	1	Sets the completion methods of the receive packets, and it affects network throughput and CPU utilization. The supported methods are: • Polling - increases the CPU utilization, because the system polls the received rings for incoming packets; however, it may increase the network bandwidth since the incoming packet is handled faster. • Adaptive - combines the interrupt and polling methods dynamically, depending on traffic type and network usage. The valid values are: • 0: polling • 1: adaptive
*InterruptModeration	1	Sets the rate at which the controller moderates or delays the generation of interrupts, making it possible to optimize network throughput and CPU utilization. When disabled, the interrupt moderation of the system generates an interrupt when the packet is received. In this mode, the CPU utilization is increased at higher data rates, because the system must handle a larger number of interrupts. However, the latency is decreased, since that packet is processed more quickly. When interrupt moderation is enabled, the system accumulates interrupts and sends a single interrupt rather than a series of interrupts. An interrupt is generated after receiving 5 packets or after the passing of 10 micro seconds from receiving the first packet. The valid values are: • 0: disable • 1: enable
RxIntModeration	2	Sets the rate at which the controller moderates or delays the generation of interrupts, making it possible to optimize network throughput and CPU utilization. The default setting (Adaptive) adjusts the interrupt rates dynamically, depending on traffic type and network usage. Choosing a different setting may improve network and system performance in certain configurations. The valid values are: • 1: static • 2: adaptive The interrupt moderation count and time are configured dynamically, based on traffic types and rate.

Value Name	Default Value	Description
pkt_rate_low	150000	Sets the packet rate below which the traffic is considered as latency traffic when using adaptive interrupt moderation. The valid values are 100 up to 1000000. **Note**: This registry value is not exposed via the UI.
pkt_rate_high	170000	Sets the packet rate above which the traffic is considered as bandwidth traffic. when using adaptive interrupt moderation. The valid values are 100 up to 1000000. **Note**: This registry value is not exposed via the UI.
*RSS	1	Sets the driver to use Receive Side Scaling (RSS) mode to improve the performance of handling incoming packets. This mode allows the adapter port to utilize the multiple CPUs in a multi-core system for receiving incoming packets and steering them to their destination. RSS can significantly improve the number of transactions per second, the number of connections per second, and the network throughput. This parameter can be set to one of two values: • 1: enable (default) Sets RSS Mode. • 0: disable The hardware is configured once to use the Toeplitz hash function and the indirection table is never changed. **Note**: the I/O Acceleration Technology (IOAT) is not functional in this mode.

Value Name	Default Value	Description
TxHashDisrtibution	3	Sets the algorithm which is used to distribute the send-packets on different send rings. The adapter uses 3 methods: • 1: Size In this method only 2 Tx rings are used. The send-packets are distributed, based on the packet size. Packets that are smaller than 128 bytes use one ring, while the larger packets use the other ring. • 2: Hash In this method the adapter calculates a hash value based on the destination IP, the TCP source and the destination port. If the packet type is not IP, the packet uses ring number 0. • 3: Hash and size In this method for each hash value, 2 rings are used: one for small packets and another one for larger packets. The valid values are: • 1: size • 2: hash • 3: hash and size **Note**: This registry value is not exposed via the UI.
RxSmallPacketBypass	0	Specifies whether received small packets bypass larger packets when indicating received packet to NDIS. This mode is useful in bi-directional applications. Enabling this mode ensures that the ACK packet will bypass the regular packet and TCP/IP stack will issue the next packet more quickly. The valid values are: • 0: disable • 1: enable **Note**: This registry value is not exposed via the UI.
ReturnPacketThreshold	0	The allowed number of free received packets on the rings. Any number above it will cause the driver to return the packet to the hardware immediately. When the value is set to 0, the adapter uses 2/3 of the received ring size. The valid values are: 0 to 4096. **Note**: This registry value is not exposed via the UI.
NumTcb	16	The number of send buffers that the driver allocates for sending purposes. Each buffer is in LSO size, if LSO is enabled, or in MTU size, otherwise. The valid values are 1 up to 64. **Note**: This registry value is not exposed via the UI.

Value Name	Default Value	Description
ThreadPoll	10000	The number of cycles that should be passed without receiving any packet before the polling mechanism stops when using polling completion method for receiving. Afterwards, receiving new packets will generate an interrupt that reschedules the polling mechanism. The valid values are 0 up to 200000. **Note**: This registry value is not exposed via the UI.
AverageFactor	16	The weight of the last polling in the decision whether to continue the polling or give up when using polling completion method for receiving. The valid values are 0 up to 256. **Note**: This registry value is not exposed via the UI.
AveragePollThreshold	10	The average threshold polling number when using polling completion method for receiving. If the average number is higher than this value, the adapter continues to poll. The valid values are 0 up to 1000. **Note**: This registry value is not exposed via the UI.
ThisPollThreshold	100	The threshold number of the last polling cycle when using polling completion method for receiving. If the number of packets received in the last polling cycle is higher than this value, the adapter continues to poll The valid values are 0 up to 1000. **Note**: This registry value is not exposed via the UI.
*HeaderDataSplit	0	Enables the driver to use header data split. In this mode, the adapter uses two buffers to receive the packet. The first buffer holds the header, while the second buffer holds the data. This method reduces the cache hits and improves the performance. The valid values are: • 0: disable • 1: enable **Note**: This registry value is not exposed via the UI.

Value Name	Default Value	Description
VlanId	0	Enables packets with VlanId. It is used when no LBFO intermediate driver is used. The valid values are: • 0: disable No Vlan Id is passed. • 1-4095 Valid Vlan Id that will be passed. **Note**: This registry value is only valid for Ethernet.
TxForwardingProcessor	Automatically selected based on RSS configuration	The processor that will be used to forward the packets sent by the forwarding thread. Default is based on number of rings and number of cores on the machine. **Note**: This registry value is not exposed via the UI.
DefaultRecvRingProcessor	Automatically selected based on RSS configuration	The type of processor which will be used for the default Receive ring. This variable handles packets that are not handled by RSS. This can be non TCP/UDP packets or even UDP packets, if they are configured to use the default ring. **Note**: This registry value is not exposed via the UI.
TxInterruptProcessor	Automatically selected based on RSS configuration	The type of processor which will be used to handle the TX completions. The default is based on a number of rings and a number of cores on the machine. **Note**: This registry value is not exposed via the UI.
*NumRSSQueues	8	The maximum number of the RSS queues that the device should use. **Note**: This registry key is only in Windows Server 2012 and above.

Value Name	Default Value	Description
BlueFlame	1	The latency-critical Send WQEs to the device. When a BlueFlame is used, the WQEs are written directly to the PCI BAR of the device (in addition to memory), so that the device may handle them without having to access memory, thus shortening the execution latency. For best performance, it is recommended to use the BlueFlame when the HCA is lightly loaded. For high-bandwidth scenarios, it is recommended to use regular posting (without BlueFlame). The valid values are: • 0: disable • 1: enable **Note**: This registry value is not exposed via the UI.
*MaxRSSProcessors	8	The maximum number of RSS processors. **Note**: This registry key is only in Windows Server 2012 and above.

3.6.6 Ethernet Registry Keys

The following section describes the registry keys that are only relevant to Ethernet driver.

Value Name	Default Value	Description
RoceMaxFrameSize	1024	The maximum size of a frame (or a packet) that can be sent by the RoCE protocol (a.k.a Maximum Transmission Unit (MTU). Using larger RoCE MTU will improve the performance; however, one must ensure that the entire system, including switches, supports the defined MTU. Ethernet packet uses the general MTU value, whereas the RoCE packet uses the RoCE MTU The valid values are: • 256 • 512 • 1024 • 2048 **Note**: This registry key is supported only in Ethernet drivers.
*PriorityVLANTag	3 (Packet Priority & VLAN Enabled)	Enables sending and receiving IEEE 802.3ac tagged frames, which include: • 802.1p QoS (Quality of Service) tags for priority-tagged packets. • 802.1Q tags for VLANs. When this feature is enabled, the Mellanox driver supports sending and receiving a packet with VLAN and QoS tag.

Value Name	Default Value	Description
PromiscuousVlan	0	Specifies whether a promiscuous VLAN is enabled or not. When this parameter is set, all the packets with VLAN tags are passed to an upper level without executing any filtering. The valid values are: • 0: disable • 1: enable **Note**: This registry value is not exposed via the UI.
UseRSSForRawIP	1	The execution of RSS on UDP and Raw IP packets. In a forwarding scenario, one can improve the performance by disabling RSS on UDP or a raw packet. In such a case, the entire receive processing of these packets is done on the processor that was defined in DefaultRecvRingProcessor registry key. The valid values are: • 0: disable • 1: enable This is also relevant for IPoIB. **Note**: This registry value is not exposed via the UI.
UseRSSForUDP	1	Used to execute RSS on UDP and Raw IP packet. In forwarding scenario you can improve the performance by disable RSS on UDP or raw packet. In such a case all the receive processing of these packets is done on the processor that was defined in DefaultRecvRingProcessor registry key. The valid values are: • 0:disabled • 1: Enabled **NOTE**: This registry value is not exposed via UI.
SingleStream	0	It used to get the maximum bandwidth when using single stream traffic. When setting the registry key to enabled the driver will forward the sending packet to another CPU. This decrease the CPU utilization of the sender and allows sending in higher rate The valid values are: • 0:disabled • 1: Enabled **NOTE**: only relevant for Ethernet and IPoIB

3.6.6.1 Flow Control Options

This group of registry keys allows the administrator to control the TCP/IP traffic by pausing frame transmitting and/or receiving operations. By enabling the Flow Control mechanism, the adapters can overcome any TCP/IP issues and eliminate the risk of data loss.

Value Name	Default Value	Description
*FlowControl	3	When Rx Pause is enabled, the receiving adapter generates a flow control frame when its received queue reaches a pre-defined limit. The flow control frame is sent to the sending adapter. When TX Pause is enabled, the sending adapter pauses the transmission if it receives a flow control frame from a link partner. The valid values are: • 0: Flow control is disabled • 1: Tx Flow control is Enabled • 2: Rx Flow control is enabled • 3: Rx & Tx Flow control is enabled
PerPriRxPause	0	When Per Priority Rx Pause is configured, the receiving adapter generates a flow control frame when its priority received queue reaches a pre-defined limit. The flow control frame is sent to the sending adapter. **Notes**: • This registry value is not exposed via the UI. • RxPause and PerPriRxPause are mutual exclusive (i.e. at most, only one of them can be set).
PerPriTxPause	0	When Per Priority TX Pause is configured, the sending adapter pauses the transmission of a specific priority, if it receives a flow control frame from a link partner. **Notes**: • This registry value is not exposed via the UI. • TxPause and PerPriTxPause are mutual exclusive (i.e. at most, only one of them can be set).

3.6.6.2 VMQ Options

This section describes the registry keys that are used to control the NDIS Virtual Machine Queue (VMQ). The VMQ supports Microsoft Hyper-V network performance, and is supported on Windows Server 2008 R2 and above.

For more details about VMQ please refer to Microsoft web site,
http://msdn.microsoft.com/en-us/library/windows/hardware/ff571034(v=vs.85).aspx

Value Name	Default Value	Description
*VMQ	1	The support for the virtual machine queue (VMQ) features of the network adapter. The valid values are: • 1: enable • 0: disable
*RssOrVmqPreference	0	Specifies whether VMQ capabilities should be enabled instead of receive-side scaling (RSS) capabilities. The valid values are: • 0: Report RSS capabilities • 1: Report VMQ capabilities **Note**: This registry value is not exposed via the UI.
*VMQLookaheadSplit	1	Specifies whether the driver enables or disables the ability to split the receive buffers into lookahead and post-lookahead buffers. The valid values are: • 0: disable • 1: enable
*VMQVlanFiltering	1	Specifies whether the device enables or disables the ability to filter network packets by using the VLAN identifier in the media access control (MAC) header. The valid values are: • 0: disable • 1: enable
MaxNumVmqs	127	The number of VMQs that the device supports in parallel. This parameter can effect memory consumption of the interface, since for each VMQ, the driver creates a separate receive ring and an allocate buffer for it. In order to minimize the memory consumption, one can reduce the number of VMs that use VMQ in parallel. However, this can affect the performance. The valid values are 1 up to 127. **Note**: This registry value is not exposed via the UI.
MaxNumMacAddrFilters	127	The number of different MAC addresses that the physical port supports. This registry key affects the number of supported MAC addresses that is reported to the OS. The valid values are 1 up to 127. **Note**: This registry value is not exposed via the UI.

Value Name	Default Value	Description
MaxNumVlanFilters	127	The number of VLANs that are supported for each port. The valid values are 1 up to 127. **Note**: This registry value is not exposed via the UI.

3.6.6.3 RoCE Options

This section describes the registry keys that are used to control RoCE mode.

Value Name	Default Value	Description
roce_mode	0 - RoCE	The RoCE mode. The valid values are: • 0 - RoCE • 4 - No RoCE **Note**: The default value depends on the WinOF package used.

3.6.7 IPoIB Registry Keys

The following section describes the registry keys that are unique to IPoIB.

Value Name	Default Value	Description
GUIDMask	0	Controls the way the MAC is generated for IPoIB interface. The driver uses the 8 bytes GUID to generate 6 bytes MAC. This value should be either 0 or contain exactly 6 non-zero digits, using binary representation. Zero (0) mask indicates its default value: 0xb' 11100111. That is, to take all, except intermediate bytes of GUID to form the MAC address. In case of an improper mask, the driver uses the default one. For more details, please refer to: http://mellanox.com/related-docs/prod_software/guid2mac_checker_user_manual.txt **Note**: This registry value is not exposed via the UI.
MediumType802_3	0	Controls the way the interface is exposed to an upper level. By default, the IPoIB is exposed as an InfiniBand interface. The user can change it and cause the interface to be an Ethernet interface by setting this registry key. The valid values are: • 0 - the interface is exposed as NdisPhysicalMediumInfiniband • 1 - the interface is exposed as NdisPhysicalMedium802_3. **Note**: This registry value is not exposed via the UI.

Value Name	Default Value	Description
SaTimeout	1000	The time, in milliseconds, before retransmitting an SA query request. The valid values are 250 up to 60000.
SaRetries	10	The number of times to retry an SA query request. The valid values are 1 up to 64.
McastIgmpMldGeneralQueryInterval	3	The number of runs of the multicast monitor before a general query is initiated. This monitor runs every 30 seconds. The valid values are 1 up to 10.
LocalEndpointMaxAge	5	The maximum number of runs of the local end point DB monitor, before an unused local endpoint is removed. The endpoint age is zeroed when it is used as a source in the send flow or a destination in the receive flow. Each monitor run will increment the age of all non VMQ local endpoints. When LocalEndpointMaxAge is reached - the endpoint will be removed. The valid values are 1 up to 20. **Note**: This registry value is not exposed via the UI.
LocalEndpointMonitorInterval	60000	The time interval (in ms) between each 2 runs of the local end point DB monitor, for aging, unused local endpoints. Each run will increment the age of all non VMQ local endpoints. The valid values are 10000 up to 1200000. **Note**: This registry value is not exposed via the UI.
EnableQPR	0	Enables query path record. The valid values are: • 0 - disable • 1 - enable
McastQueryResponseInterval	2	The number of runs of the multicast monitor (which runs every 30 seconds) allowed until a response to the IGMP/MLD queries is received. If after this period a response is not received, the driver leaves the multicast group. The valid values are 1 up to 10. **Note**: This registry value is not exposed via the UI.

3.6.8 General Registry Values

This section provides information on general registry keys that affect Mellanox driver operation.

Value Name	Default Value	Description
MaxNumRssCpus	4	The number of CPUs that participate in the RSS. The Mellanox adapter can open multiple receive rings, each ring can be processed by a different processor. When RSS is disabled, the system opens a single Rx ring. The Rx ring number that is configured should be powered of two and less than the number of processors on the system. Value Type: DWORD The valid values are 1 up to number of processors on the system.
RssBaseCpu	1	The CPU number of the first CPU that the RSS can use. NDIS uses the default value of 0 for the base CPU number, however this value is configurable and can be changed. The Mellanox adapter reads this value from registry and sets it to NDIS on driver start-up. Value Type: DWORD The valid values are 0 up to the number of processors on the system.
CheckFwVersion	1	Configures the Mellanox driver to skip validation of the FW compatibility to the driver version. Skipping this check-up is not recommended and can cause unexpected behavior. It can be used for testing purposes only. Value Type: DWORD The valid values are: • 0: Don't check • 1: Check
MaximumWorkingThreads	2	The number of working threads which can work simultaneously on receive polling. By default, the Mellanox driver creates a working thread for each Rx rings if polling or adaptive receive completion is set. Value Type: DWORD The valid values are 1 up to number of Rx rings.

3.6.9 MLX BUS Registry Keys

3.6.9.1 SR-IOV Registry Keys

SR-IOV feature can be controlled, on a machine level or per device, using the same set of Registry Keys. However, only one level must be used consistently to control SR-IOV feature. If both levels were used, the per-machine level of configuration will be enforced by the driver.

Registry Keys location for machine configuration:

```
HKLM\SYSTEM\CurrentControlSet\Services\mlx4_bus\Parameters
```

Registry Keys location for device configuration:

```
HKLM\SYSTEM\CurrentControlSet\Control\Class\{4d36e97d-e325-11ce-bfc1-
08002be10318}\<nn>\Parameters
```

For more information on how to find device index nn, please refer to 3.6.1 “Finding the Index Value of the HCA,” on page 89

Key Name	Key Type	Values	Description
SriovEnable	REG_DWORD	• 0 = RoCE (default) • 1 = SR-IOV	Configures the RDMA or SR-IOV mode. **Note**: RDMA is not supported in SR-IOV mode.
SriovPort-Mode		• 0 = auto_port 1 (default) • 1 = auto_port 2 • 2 = manual	Configures the number of VFs to be enabled by the bus driver to each port. **Note**: In auto_portX mode, port X will have the number of VFs according to the burnt value in the device and the other port will have no SR-IOV and it will support native Ethernet (i.e. no RoCE). Setting this parameter to "Manual" will configure the number of VFs for each port according to the registry key MaxVFPortX. **Note**: The number of VFs can be configured both on a Mellanox bus driver level and Network Interface level (i.e using Set-NetAdapterSriov Powershell cmdlet). The number of VFs actually available to the Network Interface is the minimum value between mellanox bus driver configuration and Network Interface configuration. For example, if 8 VFs support was burnt in firmware, SriovPortMode is auto_port1, and Network Interface was allowed 32 VFs using SetNetAdapterSriov Powershell cmdlet, the actual number of VFs available to Network Interface will be 8.
MaxVFPort1 MaxVFPort2		• 16=(default)	MaxVFPort<i> The maximum number of VFs that are allowed per port. This is the number of VFs the bus driver will open when working in manual mode. **Note**: If the total number of VFs requested is larger than the number of VFs burnt in firmware, each port X(1\2) will have the number of VFs according to the following formula: (MaxVFPortX / (MaxVPort1+MaxVPort2)*number of VFs burnt in firmware.

3.6.9.2 General Registry Keys

Registry Keys location for machine configuration:.

```
HKLM\SYSTEM\CurrentControlSet\Services\mlx4_bus\Parameters
```

Key Name	Key Type	Values	Description
AllowRese-tOnError	DWORD	• 0 - disable (default) • 1 - enable	When enabled, this setting will allow an SRIOV- IB guest VM driver to grace-fully recover from a case where the hypervisor driver is stuck by resetting the guest driver. otherwise, when a hypervisor is stuck the VM will require a restart to recover. **Caution**: This setting cannot be enabled when user-space RDMA applications such as MPI are running in the VM.

3.7 Software Development Kit (SDK)

Software Development Kit (SDK) a set of development tools that allows the creation of Infini-Band applications for MLNX_VPI software package.

The SDK package contains, header files, libraries, and code examples.

To compile the examples provided with the SDK you must install Windows Driver Kit (WDK) version 8.1 and higher over Visual Studio 2013

To open the SDK package you must run the sdk.exe file and get the complete list of files. SDK package can be found under <installation_directory>\IB\SDK

It is highly recommended to program the applications over the ND API and not over the IBAL API.

3.7.1 Network Direct Interface

The Network Direct Interface (NDI) architecture provides application developers with a net-working interface that enables zero-copy data transfers between applications, kernel-bypass I/O generation and completion processing, and one-sided data transfer operations.

NDI is supported by Microsoft and is the recommended method to write InfiniBand application. NDI exposes the advanced capabilities of the Mellanox networking devices and allows applica-tions to leverage advances of InfiniBand.

For further information please refer to:

http://msdn.microsoft.com/en-us/library/cc904397(v=vs.85).aspx

3.8 Performance Tuning and Counters

For further information on WinOF performance, please refer to the Performance Tuning Guide for Mellanox Network Adapters.

This section describes how to modify Windows registry parameters in order to improve performance.

Please note that modifying the registry incorrectly might lead to serious problems, including the loss of data, system hang, and you may need to reinstall Windows. As such it is recommended to back up the registry on your system before implementing recommendations included in this section. If the modifications you apply lead to serious problems, you will be able to restore the original registry state. For more details about backing up and restoring the registry, please visit www.microsoft.com.

3.8.1 General Performance Optimization and Tuning

To achieve the best performance for Windows, you may need to modify some of the Windows registries.

3.8.1.1 Registry Tuning

The registry entries that may be added/changed by this "General Tuning" procedure are:

Under HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters:

- Disable TCP selective acks option for better cpu utilization:

```
SackOpts, type REG_DWORD, value set to 0.
```

Under HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\AFD\Parameters:

- Enable fast datagram sending for UDP traffic:

```
FastSendDatagramThreshold, type REG_DWORD, value set to 64K.
```

Under HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Ndis\Parameters:

- Set RSS parameters:

```
RssBaseCpu, type REG_DWORD, value set to 1.
```

3.8.1.2 Enable RSS

Enabling Receive Side Scaling (RSS) is performed by means of the following command:

```
"netsh int tcp set global rss = enabled"
```

3.8.1.3 Tuning the IPoIB Network Adapter

The IPoIB Network Adapter tuning can be performed either during installation by modifying some of Windows registries as explained in Section 3.8.1.1, "Registry Tuning", on page 110. or can be set post-installation manually.

➢ ***To improve the network adapter performance, activate the performance tuning tool as follows:***

Step 1. Start the "Device Manager" (open a command line window and enter: devmgmt.msc).

Step 2. Open "Network Adapters".

Step 3. Select Mellanox IPoIB adapter, right click and select Properties.

Step 4. Select the "Performance tab".

Step 5. Choose one of the tuning scenarios:

- Single port traffic - Improves performance for running single port traffic each time.
- Dual port traffic - Improves performance for running traffic on both ports simultaneously.
- Forwarding traffic - Improves performance for running scenarios that involve both ports (for example: via IXIA)
- Multicast traffic - Improves performance when the main traffic runs on multicast.

Step 6. Click on “Run Tuning” button.

Clicking the “Run Tuning” button changes several registry entries (described below), and checks for system services that may decrease network performance. It also generates a log including the applied changes.

Users can view this log to restore the previous values. The log path is:

```
%HOMEDRIVE%\Windows\System32\LogFiles\PerformanceTunning.log
```

This tuning is required to be performed only once after the installation is completed, and on one adapter only (as long as these entries are not changed directly in the registry, or by some other installation or script).

A reboot may be required for the changes to take effect.

3.8.1.4 Tuning the Ethernet Network Adapter

The Ethernet Network Adapter general tuning can be performed during installation by modifying some of Windows registries as explained in section "Registry Tuning" on page 32. Specific scenarios tuning can be set post-installation manually.

➢ ***To improve the network adapter performance, activate the performance tuning tool as follows:***

Step 1. Start the "Device Manager" (open a command line window and enter: devmgmt.msc).

Step 2. Open "Network Adapters".

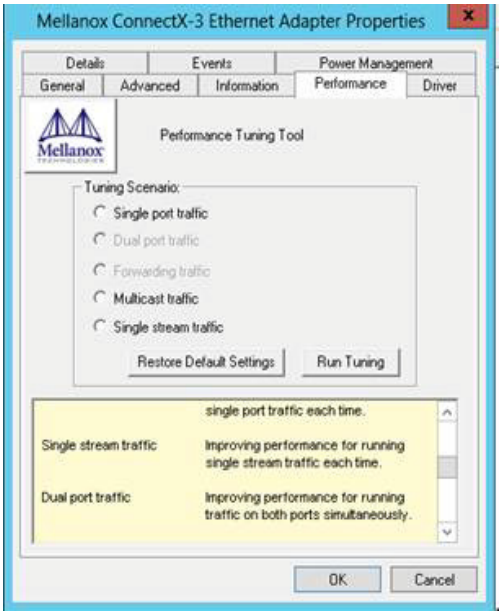
Step 3. Select Mellanox Ethernet adapter, right click and select Properties.

Step 4. Select the "Performance tab".

Step 5. Choose one of the tuning scenarios:

- Single port traffic - Improves performance for running single port traffic each time.
- Single stream traffic - Optimizes tuning for applications with single connection.
- Dual port traffic - Improves performance for running traffic on both ports simultaneously.
- Forwarding traffic - Improves performance for running scenarios that involve both ports (for example: via IXIA)
- Multicast traffic - Improves performance when the main traffic runs on multicast.

5. Click on “Run Tuning” button.



Clicking the "Run Tuning" button activates the general tuning as explained above and changes several driver registry entries for the current adapter and its sibling device once the sibling is an Ethernet device as well. It also generates a log including the applied changes.

Users can view this log to restore the previous values. The log path is:

```
%HOMEDRIVE%\Windows\System32\LogFiles\PerformanceTunning.log
```

This tuning is required to be performed only once after the installation is completed, and on one adapter only (as long as these entries are not changed directly in the registry, or by some other installation or script).

Please note that a reboot may be required for the changes to take effect.

3.8.1.4.1 Performance Tuning Tool Application

You can also activate the performance tuning through a script called perf_tuning.exe. This script has 4 options, which include the 3 scenarios described above and an additional manual tuning through which you can set the RSS base and number of processors for each Ethernet adapter. The adapters you wish to tune are supplied to the script by their name according to the "Network Connections".

Synopsis

```
perf_tuning.exe -s -c1 <first connection name> [-c2 <second connection
name>]
perf_tuning.exe -d -c1 <first connection name> -c2 <second connection name>
perf_tuning.exe -f -c1 <first connection name> -c2 <second connection name>
perf_tuning.exe -m -c1 <first connection name> -b <base RSS processor num-
ber> -n <number of RSS processors>
perf_tuning -st -c1 <first connection name> [-c2 <second connection name>]
```

Options

Flag	Description
-s	Single port traffic scenario. This option can be followed by one or two connection names. The tuning will restore the default settings on the second connection and performed on the first connection. This option automatically sets: • SendCompletionMethod = 0 • RecvCompletionMethod = 2 • *ReceiveBuffers = 1024 • In Operating Systems support NDIS6.3: RssProfile = 4 Additionally, this option chooses the best processors to assign to: • DefaultRecvRingProcessor • TxInterruptProcessor • TxForwardingProcessor • In Operating Systems support NDIS6.2: RssBaseProcNumber MaxRssProcessors • In Operating Systems support NDIS6.3: NumRSSQueues RssMaxProcNumber

Flag	Description
-d	Dual port traffic scenario. This option must be followed by two connection names. The tuning in this case is code-pendent. This option automatically sets: • SendCompletionMethod = 0 • RecvCompletionMethod = 2 • *ReceiveBuffers = 1024 • In Operating Systems support NDIS6.3: RssProfile = 4 Additionally, this option chooses the best processors to assign to: • DefaultRecvRingProcessor • TxForwardingProcessor • In Operating Systems support NDIS6.2: RssBaseProcNumber MaxRssProcessors • In Operating Systems support NDIS6.3: NumRSSQueues RssMaxProcNumber
-f	Forwarding traffic scenario. This option must be followed by two connection names. The tuning in this case is code-pendent. This option automatically sets: • SendCompletionMethod = 1 • RecvCompletionMethod = 0 • *ReceiveBuffers = 4096 • UseRSSForRawIP = 0 • UseRSSForUDP = 0 Additionally, this option chooses the best processors to assign to: • DefaultRecvRingProcessor • TxInterruptProcessor • TxForwardingProcessor • In Operating Systems support NDIS6.2: RssBaseProcNumber MaxRssProcessors • In Operating Systems support NDIS6.3: NumRSSQueues RssMaxProcNumber
-m	Manual configuration This option must be followed by one connection name. This option assigns the provided base and number of CPUs to: • *RssBaseProcNumber • *MaxRssProcessors Additionally, this option assigns the following with processors inside the range: • DefaultRecvRingProcessor • TxInterruptProcessor

Flag	Description
-r	Restore default settings. This option can be followed by one or two connection names. This option automatically sets the driver registry values back to their default values: • SendCompletionMethod = 0 - IPoIB; 1 - ETH • RecvCompletionMethod = 2 • *ReceiveBuffers = 1024 • UseRSSForRawIP = 1 • DefaultRecvRingProcessor = -1 • TxInterruptProcessor = -1 • TxForwardingProcessor = -1 • UseRSSForUDP = 1 • In Operating Systems support NDIS6.2: MaxRssProcessors = 8 • In Operating Systems support NDIS6.3: NumRSSQueues = 8
-c1	Specifies first connection name. See examples
-c2	Specifies second connection name. See examples
-b	Specifies base RSS processor number. See examples. Used for manual option (-m) only.
-n	Specifies number of RSS processors. See examples. Used for manual option (-m) only.
-st	Single stream traffic scenario. This option must be followed by one or two connection names for an Ethernet adapter. The tuning will restore the default settings on the second connection and performed on the first connection. This option automatically sets: • SendCompletionMethod = 0 • RecvCompletionMethod = 2 • *ReceiveBuffers = 1024 • In Operating Systems support NDIS6.3: RssProfile = 4 • Additionally, this option chooses the best processors to assign to: • DefaultRecvRingProcessor • TxInterruptProcessor • TxForwardingProcessor • In Operating Systems support NDIS6.2: RssBaseProcNumber MaxRssProcessors • In Operating Systems support NDIS6.3: NumRSSQueues RssMaxProcNumber

Examples

For example, if the adapter is represented by "Local Area Connection 6" and "Local Area Connection 7"

```
For single port stream tuning type:
perf_tuning.exe -s -c1 "Local Area Connection 6" -c2 "Local Area Connection
7"
or to set one adapter only:
perf_tuning.exe -s -c1 "Local Area Connection 6"
For single stream tuning type:
perf_tuning.exe -st -c1 "Local Area Connection 6" -c2 "Local Area Connection
7"
or to set one adapter only:
perf_tuning.exe -st -c1 "Local Area Connection 6"
For dual port streams tuning type:
perf_tuning.exe -d -c1 "Local Area Connection 6" -c2 "Local Area Connection
7"
For forwarding streams tuning type:
perf_tuning.exe -f -c1 "Local Area Connection 6" -c2 "Local Area Connection
7"
For manual tuning of the first adapter to use RSS on CPUs 0-3:
perf_tuning.exe -m -c1 "Local Area Connection 6" -b 0 -n 4
In order to restore defaults type:
perf_tuning.exe -r -c1 "Local Area Connection 6" -c2 "Local Area Connection
7"
```

3.8.1.5 SR-IOV Tuning

To achieve best performance on SR-IOV VF, please run the following powershell commands on the host:

```
Set-VMNetworkAdapter -Name "Network Adapter" -VMName vm1 -IovQueuePairsRequested 4
OR
Set-VMNetworkAdapter -Name "Network Adapter" -VMName vm1 -IovQueuePairsRequested 8
for 40GbE
```

3.8.1.6 Improving Live Migration

In order to improve live migration over SMB direct performance, please set the following registry key to 0 and reboot the machine:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\LanmanServer\Parameters\RequireSe-
curitySignature
```

For further details, please refer to:

http://blogs.technet.com/b/josebda/archive/2010/12/01/the-basics-of-smb-signing-covering-both-smb1-and-smb2.aspx

3.8.2 Application Specific Optimization and Tuning

3.8.2.1 Ethernet Performance Tuning

The user can configure the Ethernet adapter by setting some registry keys. The registry keys may affect Ethernet performance.

➢ ***To improve performance, activate the performance tuning tool as follows:***

Step 1. Start the "Device Manager" (open a command line window and enter: devmgmt.msc).

Step 2. Open "Network Adapters".

Step 3. Right click the relevant Ethernet adapter and select Properties.

Step 4. Select the "Advanced" tab

Step 5. Modify performance parameters (properties) as desired.

3.8.2.1.1 Performance Known Issues

- On Intel I/OAT supported systems, it is highly recommended to install and enable the latest I/OAT driver (download from www.intel.com).
- With I/OAT enabled, sending 256-byte messages or larger will activate I/OAT. This will cause a significant latency increase due to I/OAT algorithms. On the other hand, throughput will increase significantly when using I/OAT.

3.8.2.2 IPoIB Performance Tuning

The user can configure the IPoIB adapter by setting some registry keys. The registry keys may affect IPoIB performance.

For the complete list of registry entries that may be added/changed by the performance tuning procedure, see MLNX_VPI_WinOF Registry Keys following the path below:

http://www.mellanox.com/page/products_dyn?product_family=32&mtag=windows_sw_drivers

➢ ***To improve performance, activate the performance tuning tool as follows:***

Step 1. Start the "Device Manager" (open a command line window and enter: devmgmt.msc).

Step 2. Open "Network Adapters".

Step 3. Right click the relevant IPoIB adapter and select Properties.

Step 4. Select the "Advanced" tab

Step 5. Modify performance parameters (properties) as desired.

3.8.3 Tunable Performance Parameters

The following is a list of key parameters for performance tuning.

- **Jumbo Packet**

  The maximum available size of the transfer unit, also known as the Maximum Transmission Unit (MTU). For IPoIB, the MTU should not include the size of the IPoIB header (=4B). For example, if the network adapter card supports a 4K MTU, the upper threshold for payload MTU is 4092B and not 4096B. The MTU of a network can have a substantial impact on performance. A 4K MTU size improves performance for short messages, since it allows the OS to coalesce many small messages into a large one.

  - Valid MTU values range for an Ethernet driver is between 614 and 9614.
  - Valid MTU values range for an IPoIB driver is between 1500 and 4092.

All devices on the same physical network, or on the same logical network, must have the same MTU.

- **Receive Buffers**

  The number of receive buffers (default 1024).

- **Send Buffers**

  The number of sent buffers (default 2048).

- **Performance Options**

  Configures parameters that can improve adapter performance.

  - Interrupt Moderation

    Moderates or delays the interrupts' generation. Hence, optimizes network throughput and CPU utilization (default Enabled).

    - When the interrupt moderation is enabled, the system accumulates interrupts and sends a single interrupt rather than a series of interrupts. An interrupt is generated after receiving 5 packets or after 10ms from the first packet received. It improves performance and reduces CPU load however, it increases latency.
    - When the interrupt moderation is disabled, the system generates an interrupt each time a packet is received or sent. In this mode, the CPU utilization data rates increase, as the system handles a larger number of interrupts. However, the latency decreases as the packet is handled faster.

  - Receive Side Scaling (RSS Mode)

    Improves incoming packet processing performance. RSS enables the adapter port to utilize the multiple CPUs in a multi-core system for receiving incoming packets and steering them to the designated destination. RSS can significantly improve the number of transactions, the number of connections per second, and the network throughput.

    This parameter can be set to one of the following values:

    - Enabled (default): Set RSS Mode
    - Disabled: The hardware is configured once to use the Toeplitz hash function, and the indirection table is never changed.

    IOAT is not used while in RSS mode.

    - Receive Completion Method

      Sets the completion methods of the received packets, and can affect network throughput and CPU utilization.

      - **Polling Method**

        Increases the CPU utilization as the system polls the received rings for the incoming packets. However, it may increase the network performance as the incoming packet is handled faster.

      - **Interrupt Method**

        Optimizes the CPU as it uses interrupts for handling incoming messages. However, in certain scenarios it can decrease the network throughput.

      - **Adaptive (Default Settings)**

        A combination of the interrupt and polling methods dynamically, depending on traffic type and network usage. Choosing a different setting may improve network and/or system performance in certain configurations.

  - Interrupt Moderation RX Packet Count

Number of packets that need to be received before an interrupt is generated on the receive side (default 5).

- Interrupt Moderation RX Packet Time

  Maximum elapsed time (in usec) between the receiving of a packet and the generation of an interrupt, even if the moderation count has not been reached (default 10).

- Rx Interrupt Moderation Type

  Sets the rate at which the controller moderates or delays the generation of interrupts making it possible to optimize network throughput and CPU utilization. The default setting (Adaptive) adjusts the interrupt rates dynamically depending on the traffic type and network usage. Choosing a different setting may improve network and system performance in certain configurations.

- Send completion method

  Sets the completion methods of the Send packets and it may affect network throughput and CPU utilization.

- Interrupt Moderation TX Packet Count

  Number of packets that need to be sent before an interrupt is generated on the send side (default 0).

- Interrupt Moderation TX Packet Time

  Maximum elapsed time (in usec) between the sending of a packet and the generation of an interrupt even if the moderation count has not been reached (default 0).

- **Offload Options**

  Allows you to specify which TCP/IP offload settings are handled by the adapter rather than the operating system.

  Enabling offloading services increases transmission performance as the offload tasks are performed by the adapter hardware rather than the operating system. Thus, freeing CPU resources to work on other tasks.

  - IPv4 Checksums Offload

    Enables the adapter to compute IPv4 checksum upon transmit and/or receive instead of the CPU (default Enabled).

  - TCP/UDP Checksum Offload for IPv4 packets

    Enables the adapter to compute TCP/UDP checksum over IPv4 packets upon transmit and/or receive instead of the CPU (default Enabled).

  - TCP/UDP Checksum Offload for IPv6 packets

    Enables the adapter to compute TCP/UDP checksum over IPv6 packets upon transmit and/or receive instead of the CPU (default Enabled).

  - Large Send Offload (LSO)

    Allows the TCP stack to build a TCP message up to 64KB long and sends it in one call down the stack. The adapter then re-segments the message into multiple TCP packets for transmission on the wire with each pack sized according to the MTU. This option offloads a large amount of kernel processing time from the host CPU to the adapter.

- **IB Options**

  Configures parameters related to InfiniBand functionality.

  - SA Query Retry Count

    Sets the number of SA query retries once a query fails. The valid values are 1 - 64 (default 10).

- SA Query Timeout

  Sets the waiting timeout (in millisecond) of an SA query completion. The valid values are 500 - 60000 (default 1000 ms).

3.8.4 Adapter Proprietary Performance Counters

Proprietary Performance Counters are used to provide information on Operating System, application, service or the drivers' performance. Counters can be used for different system debugging purposes, help to determine system bottlenecks and fine-tune system and application performance. The Operating System, network, and devices provide counter data that the application can consume to provide users with a graphical view of the system's performance quality. WinOF counters hold the standard Windows CounterSet API that includes:

- Network Interface
- RDMA activity
- SMB Direct Connection

3.8.4.1 Supported Standard Performance Counters

3.8.4.1.1 Proprietary Mellanox Adapter Traffic Counters

Proprietary Mellanox adapter traffic counter set consists of global traffic statistics which gather information from ConnectX®-3 and ConnectX®-3 Pro network adapters, and includes traffic statistics, and various types of error and indications from both the Physical Function and Virtual Function.

Table 11 - Mellanox Adapter Traffic Counters

Mellanox Adapter Traffic Counters	Description
Bytes IN	
Bytes Received	Shows the number of bytes received by the adapter. The counted bytes include framing characters.
Bytes Received/Sec	Shows the rate at which bytes are received by the adapter. The counted bytes include framing characters.
Packets Received	Shows the number of packets received by ConnectX-3 and ConnectX-3Pro network interface.
Packets Received/Sec	Shows the rate at which packets are received by ConnectX-3 and ConnectX-3Pro network interface.
Bytes/ Packets OUT	
Bytes Sent	Shows the number of bytes sent by the adapter. The counted bytes include framing characters.
Bytes Sent/Sec	Shows the rate at which bytes are sent by the adapter. The counted bytes include framing characters.
Packets Sent	Shows the number of packets sent by ConnectX-3 and ConnectX-3Pro network interface.

Table 11 - Mellanox Adapter Traffic Counters

Mellanox Adapter Traffic Counters	Description
Packets Sent/Sec	Shows the rate at which packets are sent by ConnectX-3 and ConnectX-3Pro network interface.
Bytes' TOTAL	
Bytes Total	Shows the total of bytes handled by the adapter. The counted bytes include framing characters.
Bytes Total/Sec	Shows the total rate of bytes that are sent and received by the adapter. The counted bytes include framing characters.
Packets Total	Shows the total of packets handled by ConnectX-3 and ConnectX-3Pro network interface.
Packets Total/Sec	Shows the rate at which packets are sent and received by ConnectX-3 and ConnectX-3Pro network interface.
Control Packets	The total number of successfully received control frames
ERRORS, DROP, AND MISC. INDICATIONS	
Packets Outbound Errors	Shows the number of outbound packets that could not be transmitted because of errors.
Packets Outbound Discarded	Shows the number of outbound packets to be discarded even though no errors had been detected to prevent transmission. One possible reason for discarding packets could be to free up buffer space.
Packets Received Errors	Shows the total number of inbound packets that contained errors preventing them from being deliverable to a higher-layer protocol.
Packets Received with Frame Length Error	Shows the number of inbound packets that contained error where the frame has length error. Packets received with frame length error are a subset of packets received errors.
Packets Received with Symbol Error	Shows the number of inbound packets that contained symbol error or an invalid block. Packets received with symbol error are a subset of packets received errors.
Packets Received with Bad CRC Error	Shows the number of inbound packets that failed the CRC check. Packets received with bad CRC error are a subset of packets received errors.
Packets Received Discarded	Shows the number of inbound packets that were chosen to be discarded even though no errors had been detected to prevent their being deliverable to a higher-layer protocol. One possible reason for discarding such a packet could be to free up buffer space.

3.8.4.1.2 Proprietary Mellanox Adapter Diagnostics Counters

Proprietary Mellanox adapter diagnostics counter set consists of the NIC diagnostics. These counters collect information from ConnectX®-3 and ConnectX®-3 Pro firmware flows.

Table 12 - Mellanox Adapter Diagnostics Counters

Mellanox Adapter Diagnostics Counters	Description
Requester length errors	Number of local length errors when the local machine generates outbound traffic.
Responder length errors	Number of local length errors when the local machine receives inbound traffic.
Requester QP operation errors	Number of local QP operation errors when the local machine generates outbound traffic.
Responder QP operation errors	Number of local QP operation errors when the local machine receives inbound traffic.
Requester protection errors	Number of local protection errors when the local machine generates outbound traffic.
Responder protection errors	Number of local protection errors when the local machine receives inbound traffic.
Requester CQE errors	Number of local CQE with errors when the local machine generates outbound traffic.
Responder CQE errors	Number of local CQE with errors when the local machine receives inbound traffic.
Requester Invalid request errors	Number of remote invalid request errors when the local machine generates outbound traffic, i.e. NAK was received indicating that the other end detected invalid OpCode request.
Responder Invalid request errors	Number of remote invalid request errors when the local machine receives inbound traffic.
Requester Remote access errors	Number of remote access errors when the local machine generates outbound traffic, i.e. NAK was received indicating that the other end detected wrong rkey.
Responder Remote access errors	Number of remote access errors when the local machine receives inbound traffic, i.e. the local machine received RDMA request with wrong rkey.
Requester RNR NAK	Number of RNR (Receiver Not Ready) NAKs received when the local machine generates outbound traffic.
Responder RNR NAK	Number of RNR (Receiver Not Ready) NAKs sent when the local machine receives inbound traffic.

Table 12 - Mellanox Adapter Diagnostics Counters

Mellanox Adapter Diagnostics Counters	Description
Requester out of order sequence NAK	Number of Out of Sequence NAK received when the local machine generates outbound traffic, i.e. the number of times the local machine received NAKs indicating OOS on the receiving side.
Responder out of order sequence received	Number of Out of Sequence packet received when the local machine receives inbound traffic, i.e. the number of times the local machine received messages that are not consecutive.
Requester resync	Number of resync operations when the local machine generates outbound traffic.
Responder resync	Number of resync operations when the local machine receives inbound traffic.
Requester Remote operation errors	Number of remote operation errors when the local machine generates outbound traffic, i.e. NAK was received indicating that the other end encountered an error that prevented it from completing the request.
Requester transport retries exceeded errors	Number of transport retries exceeded errors when the local machine generates outbound traffic.
Requester RNR NAK retries exceeded errors	Number of RNR (Receiver Not Ready) NAKs retries exceeded errors when the local machine generates outbound traffic.
Bad multicast received	Number of bad multicast packet received.
Discarded UD packets	Number of UD packets silently discarded on the receive queue due to lack of receives descriptor.
Discarded UC packets	Number of UC packets silently discarded on the receive queue due to lack of receives descriptor.
CQ overflows	Number of CQ overflows. **NOTE:** this value is evaluated for the entire NIC since there are cases where CQ might be associated with both ports (i.e. the value on all ports is identical).
EQ overflows	Number of EQ overflows. **NOTE:** this value is evaluated for the entire NIC since there are cases where EQ might be associated with both ports (i.e. the value on all ports is identical).
Bad doorbells	Number of bad DoorBells

Table 12 - Mellanox Adapter Diagnostics Counters

Mellanox Adapter Diagnostics Counters	Description
Responder duplicate request received (pending firmware implementation).	Number of duplicate requests received when the local machine receives inbound traffic.
Requester time out received (pending firmware implementation).	Number of time out received when the local machine generates outbound traffic.

3.8.4.1.3 Proprietary Mellanox QoS Counters

Proprietary Mellanox QoS counter set consists of flow statistics per (VLAN) priority. Each QoS policy is associated with a priority. The counter presents the priority's traffic, pause statistic

Table 13 - Mellanox Qos Counters

Mellanox Qos Counters	Description
Bytes/Packets IN	
Bytes Received	The number of bytes received that are covered by this priority. The counted bytes include framing characters (modulo 2^64).
Bytes Received/Sec	The number of bytes received per second that are covered by this priority. The counted bytes include framing characters.
Packets Received	The number of packets received that are covered by this priority (modulo 2^64).
Packets Received/Sec	The number of packets received per second that are covered by this priority.
Bytes/Packets OUT	
Bytes Sent	The number of bytes sent that are covered by this priority. The counted bytes include framing characters (modulo 2^64).
Bytes Sent/Sec	The number of bytes sent per second that are covered by this priority. The counted bytes include framing characters.
Packets Sent	The number of packets sent that are covered by this priority (modulo 2^64).
Packets Sent/Sec	The number of packets sent per second that are covered by this priority.
Bytes and Packets Total	
Bytes Total	The total number of bytes that are covered by this priority. The counted bytes include framing characters (modulo 2^64).
Bytes Total/Sec	The total number of bytes per second that are covered by this priority. The counted bytes include framing characters.
Packets Total	The total number of packets that are covered by this priority (modulo 2^64).

Table 13 - Mellanox Qos Counters

Mellanox Qos Counters	Description
Packets Total/Sec	The total number of packets per second that are covered by this priority.
PAUSE INDICATION	
Per prio sent pause frames	The number of pause frames that were sent to priority i. The untagged instance indicates global pause that were sent.
Per prio sent pause duration	The total duration in microseconds of pause that was sent to the other end to freeze the transmission on priority i.
Per prio rcv pause frames	The number of pause frames that were received for priority i. The untagged instance indicates global pause that were received
Per prio rcv pause duration	The total duration in microseconds of pause that was requested by the other end to freeze transmission on priority i.

3.8.4.1.4 Propriety RDMA Activity

Proprietary RDMA Activity counter set consists of NDK performance counters. These performance counters allow you to track Network Direct Kernel (RDMA) activity, including traffic rates, errors, and control plane activity.

Table 14 - RDMA Activity

RDMA Activity Counters	Description
RDMA Accepted Connections	The number of inbound RDMA connections established.
RDMA Active Connections	The number of active RDMA connections.
RDMA Completion Queue Errors	This counter is not supported, and always is set to zero.
RDMA Connection Errors	The number of established connections with an error before a consumer disconnected the connection.
RDMA Failed Connection Attempts	The number of inbound and outbound RDMA connection attempts that failed.
RDMA Inbound Bytes/sec	The number of bytes for all incoming RDMA traffic. This includes additional layer two protocol overhead.
RDMA Inbound Frames/sec	The number, in frames, of layer two frames that carry incoming RDMA traffic.
RDMA Initiated Connections	The number of outbound connections established.

Table 14 - RDMA Activity

RDMA Activity Counters	Description
RDMA Outbound Bytes/sec	The number of bytes for all outgoing RDMA traffic. This includes additional layer two protocol overhead.
RDMA Outbound Frames/sec	The number, in frames, of layer two frames that carry outgoing RDMA traffic.

4 Utilities

4.1 Snapshot Tool

The snapshot tool scans the machine and provide information on the current settings of the operating system, networking and hardware.

It is highly recommended to add this report when you contact the support team

4.1.1 Snapshot Usage

The snapshot tool can be found at:

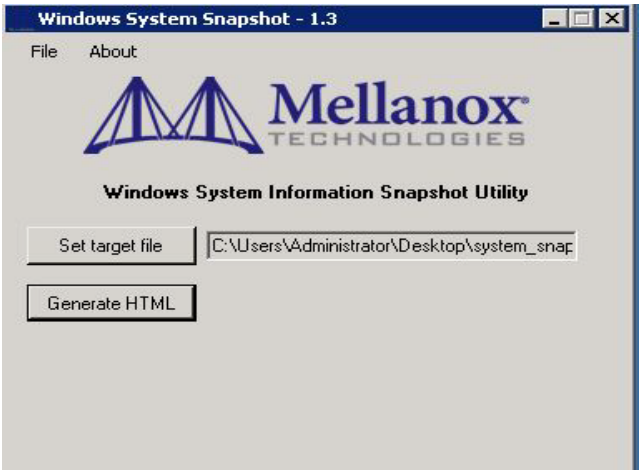
```
<installation_directory>\tools\MLNX_System_Snapshot.exe
```

The user can set the report location.

➢ ***To generate the snapshot report:***

Step 1. [Optional] Change the location of the generated file by setting the full path of the file to be generated or by pressing "Set target file" and choosing the directory that will hold the generated file and its file name.

Step 2. Click on Generate HTML button



Once the report is ready the folder which contains the report will be opened automatically.

4.2 part_man - Virtual IPoIB Port Creation Utility

part_man is used to add/remove/show virtual IPoIB ports. Each Mellanox IPoIB port can have multiple virtual IPoIB ports, which can use the default PKey value (0xffff) or a non-default value supplied by the user.

➢ ***Usage***

```
part_man.exe [-v] <show|add|rem|remall> ["Local area connection
#"]  [name] [pkey]
```

- -v: Increases verbosity level.

- show: Shows the currently configured virtual ipoib ports, along with PKey info.
- add: Adds new virtual IPoIB port. Where add should be used with interface name, as it appears in Network connection in the control panel.
- ["Local area connection #"] parameter is the interface name from the Network section of Control Panel.
- [name] parameter: Any printable name without quotation marks (“ ”), commas, and starting with i.
- rem: Removes existing virtual IPoIB port. Get the port info with the 'show' command to pass as parameters. (Note: for interfaces using default PKey, the [pkey] parameter can be excluded).
- remall: Removes all virtual IPoIB ports.

➢ ***Example***

Adding and removing virtual port with default PKey:

```
> part_man add "Ethernet 4" ipoib_4_1
Done...
> part_man show
Ethernet 6 ipoib_4_1 FFFF
> part_man rem "Ethernet 6"  ipoib_4_1
Done
```

Adding and removing virtual port with non-default PKey:

```
> part_man add "Ethernet 5" ipoib_5_1 7123
Done...

> part_man add "Ethernet 5" ipoib_5_1 F123
Done...

> part_man show
Ethernet 7 ipoib_5_1 F123

> part_man rem "Ethernet 7" ipoib_5_1 F123
Done…
```

Adding a partial membership PKey value with the upper bit turned off:

```
> part_man add "Ethernet 5" ipoib_5_1 7123
Done...
```

The new port will use the partial PKey only in absence of a full membership PKey of the same value (0xf123 for the example above) in the OpenSM configuration. Otherwise the full membership PKey will be chosen.

Make sure that the PKeys used in the part_man commands are supported by the OpenSM running on this port and the membership type of them is consistent to the one defined by OpenSM. If the PKeys are not supported the new vIPoIB port will stay in a disconnected state until the configuration is fixed.

For further details about partitions configurations for OpenSM, please refer to section 8.4 “Partitions” in MLNX OFED User Manual.

For further details about pre and post configurations for the new vIPoIB port, please refer to 3.3.1.5 “Multiple Interfaces over non-default PKeys Support,” on page 63

4.3 Vea_man- Virtual Ethernet

vea_man is a set of commands allows you to add or remove a VEA, or query the existing Mellanox ethernet adapters and see which are virtual and which are physical.

4.3.1 Adding a New Virtual Adapter

➢ ***To add a new virtual adapter, run the following command:***

```
> vea_man -a <adapter name>
```

<adapter name> is the name of the existing physical adapter which will be, essentially, cloned. The new adapter will be named by system default rules.

4.3.2 Removing a Virtual Ethernet Adapter

➢ ***To remove a virtual ethernet adapter, run the following command:***

```
> vea_man -r <adapter name>
```

4.3.3 Querying the Virtual Ethernet Database

Querying the virtual ethernet database reports all physical and virtual ethernet adapters on all Mellanox cards in the system.

➢ ***To query the virtual ethernet database, run the following command:***

```
> vea_man -q
> vea_man
```

4.3.4 Help Message

➢ ***To view the help message, run the following command:***

```
> vea_man -?
> vea_man -h
```

If your adapter name has spaces in it, you need to surround it with quotes.
Examples:
> vea_man -a "Ethernet 9" - Adds a new adapter as a virtual duplicate of Ethernet 9
> vea_man -r "Ethernet 13" - Removes virtual ethernet adapter Ethernet 13

4.4 InfiniBand Fabric Diagnostic Utilities

The diagnostic utilities described in this chapter provide means for debugging the connectivity and status of InfiniBand (IB) devices in a fabric.

4.4.1 Utilities Usage: Common Configuration, Interface and Addressing

This section first describes common configuration, interface, and addressing for all the tools in the package. Then it provides detailed descriptions of the tools themselves including: operation, synopsis and options descriptions, error codes, and examples.

Topology File (Optional)

An InfiniBand fabric is composed of switches and channel adapter (HCA/TCA) devices. To identify devices in a fabric (or even in one switch system), each device is given a GUID (a MAC equivalent). Since a GUID is a non-user-friendly string of characters, it is better to alias it to a meaningful, user-given name. For this objective, the IB Diagnostic Tools can be provided with a "topology file", which is an optional configuration file specifying the IB fabric topology in user-given names.

For diagnostic tools to fully support the topology file, the user may need to provide the local system name (if the local hostname is not used in the topology file).

To specify a topology file to a diagnostic tool use one of the following two options:

1. On the command line, specify the file name using the option '`-t <topology file name>`'
2. Define the environment variable IBDIAG_TOPO_FILE

To specify the local system name to a diagnostic tool, use one of the following two options:

1. On the command line, specify the system name using the option '`-s <local system name>`'
2. Define the environment variable IBDIAG_SYS_NAME

IB Interface Definition

The diagnostic tools installed on a machine connect to the IB fabric by means of an HCA port through which they send MADs. To specify this port to an IB diagnostic tool use one of the following options:

1. On the command line, specify the port number using the option '-p <local port number>' (see below)
2. Define the environment variable IBDIAG_PORT_NUM

In case more than one HCA device is installed on the local machine, it is necessary to specify the device's index to the tool as well. For this use on of the following options:

1. On the command line, specify the index of the local device using the following option: '-i <index of local device>'

Define the environment variable IBDIAG_DEV_IDX

Addressing

This section applies to the `ibdiagpath` tool only. A tool command may require defining the destination device or port to which it applies.

The following addressing modes can be used to define the IB ports:

- Using a Directed Route to the destination: (Tool option '-d')

  This option defines a directed route of output port numbers from the local port to the destination.

- Using port LIDs: (Tool option '-l'):

  In this mode, the source and destination ports are defined by means of their LIDs. If the fabric is configured to allow multiple LIDs per port, then using any of them is valid for defining a port.

- Using port names defined in the topology file: (Tool option '-n')

  This option refers to the source and destination ports by the names defined in the topology file. (Therefore, this option is relevant only if a topology file is specified to the tool.) In this mode, the tool uses the names to extract the port LIDs from the matched topology, then the tool operates as in the '-l' option.

For further information on the following tools, please refer to the tool's man page.

Table 15 - Diagnostic Utilities

Utility	Description
ibdiagnet	Scans the fabric using directed route packets and extracts all the available information regarding its connectivity and devices.
ibportstate	Enables querying the logical (link) and physical port states of an InfiniBand port. It also allows adjusting the link speed that is enabled on any InfiniBand port. If the queried port is a switch port, then ibportstate can be used to • Disable, enable or reset the port • Validate the port's link width and speed against the peer port
ibroute	Uses SMPs to display the forwarding tables for unicast (LinearForwardingTable or LFT) or multicast (MulticastForwardingTable or MFT) for the specified switch LID and the optional lid (mlid) range. The default range is all valid entries in the range of 1 to FDBTop.

Table 15 - Diagnostic Utilities

Utility	Description
ibdump	Dumps InfiniBand, Ethernet and all RoCE versions' traffic that flows to and from Mellanox ConnectX®-3/ConnectX®-3 Pro NIC's ports. It provides a similar functionality to the tcpdump tool on a 'standard' Ethernet port. The ibdump tool generates packet dump file in .pcap format. This file can be loaded by the Wireshark tool (www.wireshark.org) for graphical traffic analysis. This provides the ability to analyze network behavior and performance, and to debug applications that send or receive RDMA network traffic. Run "ibdump -h" to display a help message which details the tools options.
smpquery	Provides a basic subset of standard SMP queries to query Subnet management attributes such as node info, node description, switch info, and port info.
perfquery	Queries InfiniBand ports' performance and error counters. Optionally, it displays aggregated counters for all ports of a node. It can also reset counters after reading them or simply reset them.
ibping	Uses vendor MADs to validate connectivity between IB nodes. On exit, (IP) ping like output is shown. ibping is run as client/server, however the default is to run it as a client. Note also that in addition to ibping, a default server is implemented within the kernel.
ibnetdiscover	Performs IB subnet discovery and outputs a readable topology file. GUIDs, node types, and port numbers are displayed as well as port LIDs and NodeDescriptions. All nodes (and links) are displayed (full topology). Optionally, this utility can be used to list the current connected nodes by node-type. The output is printed to standard output unless a topology file is specified.
ibtracert	Uses SMPs to trace the path from a source GID/LID to a destination GID/LID. Each hop along the path is displayed until the destination is reached or a hop does not respond. By using the -m option, multicast path tracing can be performed between source and destination nodes.
sminfo	Optionally sets and displays the output of a sminfo query in a readable format. The target SM is the one listed in the local port info, or the SM specified by the optional SM lid or by the SM direct routed path
ibclearerrors	Clears the PMA error counters in PortCounters by either waking the InfiniBand subnet topology or using an already saved topology file.
ibstat	Displays basic information obtained from the local IB driver. Output includes LID, SMLID, port state, link width active, and port physical state.
vstat	Displays information on the HCA attributes.

Table 15 - Diagnostic Utilities

Utility	Description
osmtest	Validates InfiniBand subnet manager and administration (SM/SA). Default is to run all flows with the exception of the QoS flow. osmtest provides a test suite for opensm.
ibaddr	Displays the lid (and range) as well as the GID address of the port specified (by DR path, lid, or GUID) or the local port by default.
ibcacheedit	Allows users to edit an ibnetdiscover cache created through the --cache option in ibnetdiscover(8)
iblinkinfo	Reports link info for each port in an IB fabric, node by node. Optionally, iblinkinfo can do partial scans and limit its output to parts of a fabric.
ibqueryerrors	Reports the port error counters which exceed a threshold for each port in the fabric. The default threshold is zero (0). Error fields can also be suppressed entirely. In addition to reporting errors on every port. ibqueryerrors can report the port transmit and receive data as well as report full link information to the remote port if available.
ibsysstat	Uses vendor MADs to validate connectivity between InfiniBand nodes and obtain other information about the InfiniBand node. ibsysstat is run as client/server. Default is to run as client.
saquery	Issues the selected SA query. Node records are queried by default.
smpdump	Gets SM attributes from a specified SMA. The result is dumped in hex by default.

4.5 Fabric Performance Utilities

The performance utilities described in this chapter are intended to be used as a performance micro-benchmark. They support both InfiniBand and RoCE.

For further information on the following tools, please refer to the tool's man page.

Table 16 - Fabric Performance Utilities

Utility	Description
nd_write_bw	This test is used for performance measuring of RDMA-Write requests in Microsoft Windows Operating Systems. nd_write_bw is performance oriented for RDMA-Write with maximum throughput, and runs over Microsoft's NetworkDirect standard. The level of customizing for the user is relatively high. User may choose to run with a customized message size, customized number of iterations, or alternatively, customized test duration time. nd_write_bw runs with all message sizes from 1B to 4MB (powers of 2), message inlining, CQ moderation.
nd_write_lat	This test is used for performance measuring of RDMA-Write requests in Microsoft Windows Operating Systems. nd_write_lat is performance oriented for RDMA-Write with minimum latency, and runs over Microsoft's NetworkDirect standard. The level of customizing for the user is relatively high. User may choose to run with a customized message size, customized number of iterations, or alternatively, customized test duration time. nd_write_lat runs with all message sizes from 1B to 4MB (powers of 2), message inlining, CQ moderation.
nd_read_bw	This test is used for performance measuring of RDMA-Read requests in Microsoft Windows Operating Systems. nd_read_bw is performance oriented for RDMA-Read with maximum throughput, and runs over Microsoft's NetworkDirect standard. The level of customizing for the user is relatively high. User may choose to run with a customized message size, customized number of iterations, or alternatively, customized test duration time. nd_read_bw runs with all message sizes from 1B to 4MB (powers of 2), message inlining, CQ moderation.
nd_read_lat	This test is used for performance measuring of RDMA-Read requests in Microsoft Windows Operating Systems. nd_read_lat is performance oriented for RDMA-Read with minimum latency, and runs over Microsoft's NetworkDirect standard. The level of customizing for the user is relatively high. User may choose to run with a customized message size, customized number of iterations, or alternatively, customized test duration time. nd_read_lat runs with all message sizes from 1B to 4MB (powers of 2), message inlining, CQ moderation.
nd_send_bw	This test is used for performance measuring of Send requests in Microsoft Windows Operating Systems. nd_send_bw is performance oriented for Send with maximum throughput, and runs over Microsoft's NetworkDirect standard. The level of customizing for the user is relatively high. User may choose to run with a customized message size, customized number of iterations, or alternatively, customized test duration time. nd_send_bw runs with all message sizes from 1B to 4MB (powers of 2), message inlining, CQ moderation.

Utility	Description
nd_send_lat	This test is used for performance measuring of Send requests in Microsoft Windows Operating Systems. nd_send_lat is performance oriented for Send with minimum latency, and runs over Microsoft's NetworkDirect standard. The level of customizing for the user is relatively high. User may choose to run with a customized message size, customized number of iterations, or alternatively, customized test duration time. nd_send_lat runs with all message sizes from 1B to 4MB (powers of 2), message inlining, CQ moderation.
NTttcp	NTttcp is a Windows base testing application that sends and receives TCP data between two or more endpoints. It is a Winsock-based port of the ttcp tool that measures networking performance bytes/second. To download the latest version of NTttcp (5.28), please refer to Microsoft website following the link below: http://gallery.technet.microsoft.com/NTttcp-Version-528-Now-f8b12769 **NOTE**: This tool should be run from cmd only.

The following InfiniBand performance tests are deprecated and might be removed in future releases.

Table 17 - Deprecated Performance Utilities

Utility	Description
ib_read_bw	Calculates the BW of RDMA read between a pair of machines. One acts as a server and the other as a client. The client RDMA reads the server memory and calculate the BW by sampling the CPU each time it receive a successful completion. The test supports features such as Bidirectional, in which they both RDMA read from each other memory's at the same time, change of mtu size, tx size, number of iteration, message size and more. Read is available only in RC connection mode (as specified in IB spec).
ib_read_lat	Calculates the latency of RDMA read operation of message_size between a pair of machines. One acts as a server and the other as a client. They perform a ping pong benchmark on which one side RDMA reads the memory of the other side only after the other side have read his memory. Each of the sides samples the CPU clock each time they read the other side memory , in order to calculate latency. Read is availible only in RC connection mode (as specified in IB spec).

Utility	Description
ib_send_bw	Calculates the BW of SEND between a pair of machines. One acts as a server and the other as a client. The server receive packets from the client and they both calculate the throughput of the operation. The test supports features such as Bidirectional, on which they both send and receive at the same time, change of mtu size, tx size, number of iteration, message size and more. Using the "-a" provides results for all message sizes.
ib_send_lat	Calculates the latency of sending a packet in message_size between a pair of machines. One acts as a server and the other as a client. They perform a ping pong benchmark on which you send packet only if you receive one. Each of the sides samples the CPU each time they receive a packet in order to calculate the latency.
ib_write_bw	Calculates the BW of RDMA write between a pair of machines. One acts as a server and the other as a client. The client RDMA writes to the server memory and calculate the BW by sampling the CPU each time it receive a successful completion. The test supports features such as Bidirectional, in which they both RDMA write to each other at the same time, change of mtu size, tx size, number of iteration, message size and more. Using the "-a" flag provides results for all message sizes.
ib_write_lat	Calculates the latency of RDMA write operation of message_size between a pair of machines. One acts as a server and the other as a client. They perform a ping pong benchmark on which one side RDMA writes to the other side memory only after the other side wrote on his memory. Each of the sides samples the CPU clock each time they write to the other side memory, in order to calculate latency.
ibv_read_bw	This is a more advanced version of ib_read_bw and contains more flags and features than the older version and also improved algorithms. ibv_read_bw calculates the BW of RDMA read between a pair of machines. One acts as a server, and the other as a client. The client RDMA reads the server memory and calculate the BW by sampling the CPU each time it receive a successful completion. The test supports a large variety of features as described below, and has better performance than ib_read_bw in Nahalem systems. Read is available only in RC connection mode (as specified in the InfiniBand spec).

Utility	Description
ibv_read_lat	This is a more advanced version of ib_read_lat and contains more flags and features than the older version and also improved algorithms. ibv_read_lat calculates the latency of RDMA read operation of message_size between a pair of machines. One acts as a server and the other as a client. They perform a ping pong benchmark on which one side RDMA reads the memory of the other side only after the other side have read his memory. Each of the sides samples the CPU clock each time they read the other side memory, to calculate latency. Read is available only in RC connection mode (as specified in InfiniBand spec).
ibv_send_bw	This is a more advanced version of ib_send_bw and contains more flags and features than the older version and also improved algorithms. ibv_send_bw calculates the BW of SEND between a pair of machines. One acts as a server and the other as a client. The server receive packets from the client and they both calculate the throughput of the operation.The test supports a large variety of features as described below, and has better performance than ib_send_bw in Nehalem systems.
ibv_send_lat	This is a more advanced version of ib_send_lat and contains more flags and features than the older version and also improved algorithms. ibv_send_lat calculates the latency of sending a packet in message_size between a pair of machines. One acts as a server and the other as a client. They perform a ping pong benchmark on which you send packet only after you receive one. Each of the sides samples the CPU clock each time they receive a send packet, in order to calculate the latency.
ibv_write_bw	This is a more advanced version of ib_write_bw ,and contains more flags and features than the older version and also improved algorithms. ibv_write_bw calculates the BW of RDMA write between a pair of machines. One acts as a server and the other as a client. The client RDMA writes to the server memory and calculate the BW by sampling the CPU each time it receives a successful completion. The test supports a large variety of features as described below, and has better performance than ib_write_bw in Nehalem systems.
ibv_write_lat	This is a more advanced version of ib_write_lat and contains more flags and features than the older version and also improved algorithms. ibv_write_lat calculates the latency of RDMA write operation of message_size between a pair of machines. One acts as a server, and the other as a client. They perform a ping pong benchmark on which one side RDMA writes to the other side memory only after the other side wrote on his memory. Each of the sides samples the CPU clock each time they write to the other side memory to calculate latency.

5 Troubleshooting

You may be able to easily resolve the issues described in this section. If a problem persists and you are unable to resolve it yourself please contact your Mellanox representative or Mellanox Support at support@mellanox.com.

5.1 InfiniBand Related Troubleshooting

Table 18 - InfiniBand Related Issues

Issue	Cause	Solution
The InfiniBand interfaces are not up after the first reboot after the installation process is completed.	There is no running SM on the subnet.	To troubleshoot this issue, follow the steps below: 1. Check that the InfiniBand driver is running on all nodes by using `"vstat"`. The vstat utility located at `<installation_directory>\tools`, displays the status and capabilities of the network adaptor card(s). 2. On the command line, enter `"vstat"` (use -h for options) to retrieve information about one or more adapter ports. The field port_state will be equal to: • `PORT_DOWN` - when there is no InfiniBand cable ("no link"); • `PORT_INITIALIZED` - when the port is connected to some other port ("physical link"); • `PORT_ACTIVE` - when the port is connected and OpenSM is running ("logical link") • `PORT_ARMED` - when the port is connected to some other port ("physical link"); 3. Run "`sminfo`" and verify that OpenSM is running. In case OpenSM is not running, please see OpenSM operation instructions in Section 3.2.2, "OpenSM - Subnet Manager", on page 60 above. 4. Verify the status of ports by using vstat: All connected ports should report `"PORT_ACTIVE"` state.

5.2 Installation Related Troubleshooting

Table 19 - Installation Related Issues

Issue	Cause	Solution
The installation of WinOF fails with the following error message: `"This installation package is not supported by this processor type. Contact your product vendor."`	An incorrect driver version was installed, e.g., you are trying to install a 64-bit driver on a 32-bit machine (or vice versa).	Use the correct driver package according to the CPU architecture.
Setup fails when the remote desktop host service is installed.	A known issue in windows when using the chain MSI feature.	Disable the service before the installation and enable it at the end.

5.3 Ethernet Related Troubleshooting

Table 20 - Ethernet Related Issues

Issue	Cause	Solution
Low performance.	Non-optimal system configuration.	See section "Performance Tuning and Counters" on page 110. to take advantage of Mellanox 10/40/56 GBit NIC performance.
The driver fails to start.	An RSS configuration mismatch between the TCP stack and the Mellanox adapter.	1. Open the event log and look under `"System"` for the `"mlx4ethX"` source. 2. If found, enable RSS, run: `"netsh int tcp set global rss = enabled"`. or A less recommended suggestion as it will cause low performance. • Disable RSS on the adapter, run: `"netsh int tcp set global rss = no dynamic balancing"`.
The driver fails to start and a yellow sign appears near the `"Mellanox ConnectX 10Gb Ethernet Adapter"` in the Device Manager display.	A hardware error.	Disable and re-enable `"Mellanox ConnectX Adapter"` from the Device Manager display.

Table 20 - Ethernet Related Issues

Issue	Cause	Solution
The driver fails to start and in the Event log, under the mlx4_bus source, the following error message appears: "RUN_FW command failed with error -22"	A wrong firmware image has been programmed on the adapter card.	See Section 2.7,"Firmware Upgrade," on page 25.
No connectivity to a Fault Tolerance bundle while using network capture tools (e.g., Wireshark).	The network capture tool captures the network traffic of the non-active adapter in the bundle. This is not allowed since the tool sets the packet filter to "promiscuous", thus causing traffic to be transferred on multiple interfaces.	Close the network capture tool on the physical adapter card, and set it on the LBFO interface instead.
No Ethernet connectivity on 10Gb adapters after activating Performance Tuning (part of the installation).	A TcpWindowSize registry valuewas added.	• Remove the value key under HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\TcpWindowSize Or • Set its value to 0xFFFF.
Packets are being lost.	The port MTU has been set to a value higher than the maximum MTU supported by the switch.	Change the MTU according to the maximun MTU supported by the switch.
NVGRE changes done on a running VM, are not propagated to the VM.	The configuration changes do not take effect until the OS is restarted.	Stop the VM and afterwards perform any NVGRE configuration changes on the VM connected to the SR-IOV-enabled virtual switch.

5.4 Performance Related Troubleshooting

Table 21 - Performance Related Issues

Issue	Cause	Solution
Low performance issues	The OS profile is not configured for maximun performace.	1. Go to "Power Options" in the "Control Panel". Make sure "Maximum Performance" is set as the power scheme 2. Reboot the machine.
Flow Control is disabled when kernel debugger is configured in Windows 2012 and above.	When a kernel debugger is configured (not necessarily physically connected) then the flow control is disabled.	Set the registry key as following: `HKLM\SYSTEM\CurrentControlSet\Services\NDIS\Parameters` • Type: `REG_DWORD` • Key name: `AllowFlowControlUnderDebugger` • Value: `1`

Package drop or low performance on specific traffic class.	Lacl od QoS and Flow Control settings configuration or their misconfiguration.	Check the configured settings for all of the QoS options. Open a PowerShell prompt and use "Get-NetAdapterQos". To achieve maximum performance all of the following must exist:
		• All of the hosts, switches and routers should use the same matching flow control settings. If Global-pause is used, all devices must be configured for it. If PFC (Priority Flow-control) is used all devices must have matching settings for all priorities.
		• ETS settings that limit speed of some priorities will greatly affect the output results.
		• Make sure Flow-Control is enabled on the Mellanox Interfaces (enabled by default). Go to the device manager, right click the Mellanox interface go to "Advanced" and make sure Flow-control is enabled for both TX and RX.
		• To eliminate QoS and Flow-control as the performance degrading factor, set all devices to run with Global Pause and rerun the tests:
		  • Set Global pause on the switches, routers.
		  • Run "Disable-NetAdapterQos *" on all of the hosts in a PowerShell window.

5.5 Virtualization Related Troubleshooting

Table 22 - Virtualization Related Issues

Issue	Cause	Solution
Mellanox driver fails to load a host machine in SR-IOV environment and appears with yellow bang in Device Manager.	The device cannot find enough free resources that it can use. (Code 12).	1. Boot to BIOS and disable SR-IOV. 2. Burn Firmware with lower number of VFs. 3. Re-enable SR-IOV in BIOS. For more information, please contact Mellanox support.
Running Windows 2008 R2 and above as VM over ESX with Mellanox adpter cards connected as Direct pass-through fails to power on.	ConnectX adapter network cards are trying to use too many MSI-X verctors.	1. Go to the vSphere Web Client. 2. Right-click the virtual machine and select Edit Settings. 3. Click the Options tab and expand Advanced. 4. Click Edit Configuration. 5. Click Add Row. 6. Add the parameter to the new row: • In the Name column, add pciPassthru0.maxMSIXvectors. • In the Value column, add 31. 7. Click OK and click OK again. For further details, please refer to: http://kb.vmware.com/selfservice/microsites/search.do?cmd=displayKC&docType=kc&externalId=2032981&sliceId=1&docTypeID=DT_KB_1_1&dialogID=408420191&stateId=1 0 388456420
When enabling the VMQ, in case NVGRE offload is enabled, and a teaming of two virtual ports is performed, no ping is detected between the VMs and/or ping is detected but no establishing of TCP connection is possible.	Missing critical Microsoft updates.	Please refer to: http://support.microsoft.com/kb/2975719 "August 2014 update rollup for Windows RT 8.1, Windows 8.1, and Windows Server 2012 R2" – specifically, fixes.

Table 22 - Virtualization Related Issues

Issue	Cause	Solution
In Hyper-V environment, `Enable-NetAdapterVmq` powershell command can enable VMQ on a network adapter only if the virtual switch which does not have SR-IOV enabled is defined over corresponding network adapter.	The powershell command depends on two registry fields: `*VMQ` and `*RssOrVmqPreference`, when the former is controlled by powershell and the latter is controlled by the virtual switch.	For further information on these registry keys, please refer to: http://msdn.microsoft.com/en-us/library/windows/hardware/hh451362(v=vs.85).aspx

5.6 General Diagnostic

Issue 1. Go to “Device Manager”, locate the Mellanox adapter that you are debugging, right-click and choose “Properties” and go to “Information” tab:

- PCI Gen 2: should appear as "PCI-E 5.0 GT/s
- PCI Gen 3: should appear as "PCI-E 8.0 GT/s"
- Link Speed: 56.0 Gbps / 40.0Gbps / 10.0Gbps

Issue 2. To determine if the Mellanox NIC and PCI bus can achieve their maximum speed, it's best to run ib_send_bw in a loopback. On the same machine:

1. Run "start /b /affinity 0x1 ibv_write_bw"
2. Run "start /b /affinity 0x2 ibv_write_bw 127.0.0.1"
3. Repeat for port 2 with additional -p2, and for other cards if necessary.
4. On PCI Gen3 the expected result is around 5700MB/s

On PCI Gen2 the expected result is around 3300MB/s

Any number lower than that points to bad configuration or installation on the wrong PCI slot. Malfunctioning QoS settings and Flow Control can be the cause as well.

Issue 3. To determine the maximum speed between the two sides with the most basic test:

1. Run "ib_send_bw" on machine 1
2. Run "ib_send_bw <host1>" on machine 2 where <host1> is the hostname for machine 1.
3. Results appear in MB/s (Mega Bytes 2^20), and reflect the actual data that was transferred, excluding headers.
4. If these results are not as expected, the problem is most probably with one or more of the following:

- Old Firmware version.
- Misconfigured Flow-control: Global pause or PFC is configured wrong on the hosts, routers and-switches. See Section 3.1.4,“RDMA over Converged Ethernet (RoCE),” on page 34
- CPU/power options are not set to "Maximum Performance".

5.7 Reported Driver Events

The driver records events in the system log of the Windows event system which can be used to identify, diagnose, and predict sources of system problems.

To see the log of events, open System Event Viewer as follows:

- Right click on My Computer, click Manage, and then click Event Viewer.

  OR

1. Click start-->Run and enter "eventvwr.exe".
2. In Event Viewer, select the system log.

   The following events are recorded:

   - Mellanox ConnectX EN 10Gbit Ethernet Adapter <X> has been successfully initialized and enabled.
   - Failed to initialize Mellanox ConnectX EN 10Gbit Ethernet Adapter.
   - Mellanox ConnectX EN 10Gbit Ethernet Adapter <X> has been successfully initialized and enabled. The port's network address is <MAC Address>
   - The Mellanox ConnectX EN 10Gbit Ethernet was reset.
   - Failed to reset the Mellanox ConnectX EN 10Gbit Ethernet NIC. Try disabling then re-enabling the "Mellanox Ethernet Bus Driver" device via the Windows device manager.
   - Mellanox ConnectX EN 10Gbit Ethernet Adapter <X> has been successfully stopped.
   - Failed to initialize the Mellanox ConnectX EN 10Gbit Ethernet Adapter <X> because it uses old firmware version (<old firmware version>). You need to burn firmware version <new firmware version> or higher, and to restart your computer.
   - Mellanox ConnectX EN 10Gbit Ethernet Adapter <X> device detected that the link connected to port <Y> is up, and has initiated normal operation.
   - Mellanox ConnectX EN 10Gbit Ethernet Adapter <X> device detected that the link connected to port <Y> is down. This can occur if the physical link is disconnected or damaged, or if the other end-port is down.
   - Mismatch in the configurations between the two ports may affect the performance. When Using MSI-X, both ports should use the same RSS mode. To fix the problem, configure the RSS mode of both ports to be the same in the driver GUI.
   - Mellanox ConnectX EN 10Gbit Ethernet Adapter <X> device failed to create enough MSI-X vectors. The Network interface will not use MSI-X interrupts. This may affects the performance. To fix the problem, configure the number of MSI-X vectors in the registry to be at least <Y>.

5.8 Installation Error Codes and Troubleshooting

5.8.1 Setup Return Codes

Table 23 - Setup Return Codes

Error Code	Description	Troubleshooting
1603	Fatal error during installation	Contact support
1633	The installation package is not supported on this platform.	Make sure you are installing the right package for your platform

For additional details on Windows installer return codes, please refer to:
http://support.microsoft.com/kb/229683

5.8.2 Firmware Burning Warning Codes

Table 24 - Firmware Burning Warning Codes

Error Code	Description	Troubleshooting
1004	Failed to open the device	Contact support
1005	Could not find an image for at least one device	The firmware for your device was not found. Please try to manually burn the firmware.
1006	Found one device that has multiple images	Burn the firmware manually and select the image you want to burn.
1007	Found one device for which force update is required	Burn the firmware manually with the force flag.
1008	Found one device that has mixed versions	The firmware version or the expansion rom version does not match.

For additional details, please refer to the MFT User Manual:

http://www.mellanox.com > Products > Firmware Tools

5.8.3 Restore Configuration Warnings

Table 25 - Restore Configuration Warnings

Error Code	Description	Troubleshooting
3	Failed to restore the configu-ration	Please see log for more details and contact the support team

Appendix A: NVGRE Configuration Scripts Examples

The setup is as follow for both examples below:

```
 Hypervisor mtlae14 = "Port1", 192.168.20.114/24
      VM on mtlae14 = mtlae14-005, 172.16.14.5/16, Mac 00155D720100
      VM on mtlae14 = mtlae14-006, 172.16.14.6/16, Mac 00155D720101
 Hypervisor mtlae15 = "Port1", 192.168.20.115/24
      VM on mtlae15 = mtlae15-005, 172.16.15.5/16, Mac 00155D730100
      VM on mtlae15 = mtlae15-006, 172.16.15.6/16, Mac 00155D730101
```

A.1 Adding NVGRE Configuration to Host 14 Example

The following is an example of adding NVGRE to Host 14.

```
 # On both sides
 # vSwitch create command

 # Note, that vSwitch configuration is persistent, no need to configure it after each
reboot

 New-VMSwitch "VSwMLNX" -NetAdapterName "Port1" -AllowManagementOS $true

 # Shut down VMs
 Stop-VM -Name "mtlae14-005" -Force -Confirm
 Stop-VM -Name "mtlae14-006" -Force -Confirm
 # Connect VM to vSwitch (maybe you have to switch off VM before), doing manual does also
work
 # Connect-VMNetworkAdapter -VMName " mtlae14-005" -SwitchName "VSwMLNX"
 Add-VMNetworkAdapter -VMName "mtlae14-005" -SwitchName "VSwMLNX" -StaticMacAddress
"00155D720100"
 Add-VMNetworkAdapter -VMName "mtlae14-006" -SwitchName "VSwMLNX" -StaticMacAddress
"00155D720101"
 # -------  The commands from Step 2 - 4 are not persistent, Its suggested to create
script is running after each OS reboot

 # Step 2. Configure a Subnet Locator and Route records on each Hyper-V Host (Host 1 and
Host 2) mtlae14 & mtlae15
 New-NetVirtualizationLookupRecord -CustomerAddress 172.16.14.5 -ProviderAddress
192.168.20.114 -VirtualSubnetID 5001 -MACAddress "00155D720100" -Rule "TranslationMetho-
dEncap"
 New-NetVirtualizationLookupRecord -CustomerAddress 172.16.14.6 -ProviderAddress
192.168.20.114 -VirtualSubnetID 5001 -MACAddress "00155D720101" -Rule "TranslationMetho-
dEncap"
 New-NetVirtualizationLookupRecord -CustomerAddress 172.16.15.5 -ProviderAddress
192.168.20.115 -VirtualSubnetID 5001 -MACAddress "00155D730100" -Rule "TranslationMetho-
dEncap"
 New-NetVirtualizationLookupRecord -CustomerAddress 172.16.15.6 -ProviderAddress
192.168.20.115 -VirtualSubnetID 5001 -MACAddress "00155D730101" -Rule "TranslationMetho-
dEncap"
 # Add customer route
 New-NetVirtualizationCustomerRoute -RoutingDomainID "{11111111-2222-3333-4444-
000000005001}" -VirtualSubnetID "5001" -DestinationPrefix "172.16.0.0/16" -NextHop
"0.0.0.0" -Metric 255
```

```
 # Step 3. Configure the Provider Address and Route records on Hyper-V Host 1 (Host 1
Only) mtlae14
 $NIC = Get-NetAdapter "Port1"
 New-NetVirtualizationProviderAddress -InterfaceIndex $NIC.InterfaceIndex -Pro-
viderAddress 192.168.20.114 -PrefixLength 24
 New-NetVirtualizationProviderRoute -InterfaceIndex $NIC.InterfaceIndex -Destination-
Prefix "0.0.0.0/0" -NextHop 192.168.20.1
 # Step 5. Configure the Virtual Subnet ID on the Hyper-V Network Switch Ports for each
Virtual Machine on each Hyper-V Host (Host 1 and Host 2)
 # Run the command below for each VM on the host the VM is running on it, i.e. the for
mtlae14-005, mtlae14-006 on
 # host 192.168.20.114 and for VMs mtlae15-005, mtlae15-006 on host 192.168.20.115
 # mtlae14 only
 Get-VMNetworkAdapter -VMName mtlae14-005 | where {$_.MacAddress -eq "00155D720100"} |
Set-VMNetworkAdapter -VirtualSubnetID 5001
 Get-VMNetworkAdapter -VMName mtlae14-006 | where {$_.MacAddress -eq "00155D720101"} |
Set-VMNetworkAdapter -VirtualSubnetID 5001
```

A.2 Adding NVGRE Configuration to Host 15 Example

The following is an example of adding NVGRE to Host 15.

```
 # On both sides
 # vSwitch create command

 # Note, that vSwitch configuration is persistent, no need to configure it after each
reboot

 New-VMSwitch "VSwMLNX" -NetAdapterName "Port1" -AllowManagementOS $true

 # Shut down VMs
 Stop-VM -Name "mtlae15-005" -Force -Confirm
 Stop-VM -Name "mtlae15-006" -Force -Confirm
 # Connect VM to vSwitch (maybe you have to switch off VM before), doing manual does also
work
 # Connect-VMNetworkAdapter -VMName " mtlae14-005" -SwitchName "VSwMLNX"
 Add-VMNetworkAdapter -VMName "mtlae15-005" -SwitchName "VSwMLNX" -StaticMacAddress
"00155D730100"
 Add-VMNetworkAdapter -VMName "mtlae15-006" -SwitchName "VSwMLNX" -StaticMacAddress
"00155D730101"
```

```
 # -------  The commands from Step 2 - 4 are not persistent, Its suggested to create
script is running after each OS reboot

 # Step 2. Configure a Subnet Locator and Route records on each Hyper-V Host (Host 1 and
Host 2) mtlae14 & mtlae15
 New-NetVirtualizationLookupRecord -CustomerAddress 172.16.14.5 -ProviderAddress
192.168.20.114 -VirtualSubnetID 5001 -MACAddress "00155D720100" -Rule "TranslationMetho-
dEncap"
 New-NetVirtualizationLookupRecord -CustomerAddress 172.16.14.6 -ProviderAddress
192.168.20.114 -VirtualSubnetID 5001 -MACAddress "00155D720101" -Rule "TranslationMetho-
dEncap"
 New-NetVirtualizationLookupRecord -CustomerAddress 172.16.15.5 -ProviderAddress
192.168.20.115 -VirtualSubnetID 5001 -MACAddress "00155D730100" -Rule "TranslationMetho-
dEncap"
 New-NetVirtualizationLookupRecord -CustomerAddress 172.16.15.6 -ProviderAddress
192.168.20.115 -VirtualSubnetID 5001 -MACAddress "00155D730101" -Rule "TranslationMetho-
dEncap"
 # Add customer route
 New-NetVirtualizationCustomerRoute -RoutingDomainID "{11111111-2222-3333-4444-
000000005001}" -VirtualSubnetID "5001" -DestinationPrefix "172.16.0.0/16" -NextHop
"0.0.0.0" -Metric 255
 # Step 4. Configure the Provider Address and Route records on Hyper-V Host 2 (Host 2
Only) mtlae15
 $NIC = Get-NetAdapter "Port1"
 New-NetVirtualizationProviderAddress -InterfaceIndex $NIC.InterfaceIndex -Pro-
viderAddress 192.168.20.115 -PrefixLength 24
 New-NetVirtualizationProviderRoute -InterfaceIndex $NIC.InterfaceIndex -Destination-
Prefix "0.0.0.0/0" -NextHop 192.168.20.1
 # Step 5. Configure the Virtual Subnet ID on the Hyper-V Network Switch Ports for each
Virtual Machine on each Hyper-V Host (Host 1 and Host 2)
 # Run the command below for each VM on the host the VM is running on it, i.e. the for
mtlae14-005, mtlae14-006 on
 # host 192.168.20.114 and for VMs mtlae15-005, mtlae15-006 on host 192.168.20.115
 # mtlae15 only
 Get-VMNetworkAdapter -VMName mtlae15-005 | where {$_.MacAddress -eq "00155D730100"} |
Set-VMNetworkAdapter -VirtualSubnetID 5001
 Get-VMNetworkAdapter -VMName mtlae15-006 | where {$_.MacAddress -eq "00155D730101"} |
Set-VMNetworkAdapter -VirtualSubnetID 5001
```

Appendix B: Windows MPI (MS-MPI)

B.1 Overview

Message Passing Interface (MPI) is meant to provide virtual topology, synchronization, and communication functionality between a set of processes.

With MPI you can run one process on several hosts.

- Windows MPI run over the following protocols:
  - Sockets (Ethernet)
  - Network Direct (ND)

B.1.1 Prerequisites

- Install HPC (Build: 4.0.3906.0).
- Validate traffic (ping) between the whole MPI Hosts.
- Every MPI client need to run smpd process which open the mpi channel.
- MPI Initiator Server need to run: mpiexec. If the initiator is also client it should also run smpd.

B.2 Running MPI

Step 1. Run the following command on each mpi client.

```
start smpd -d -p <port>
```

Step 2. Install ND provider on each MPI client in MPI ND.

Step 3. Run the following command on MPI server.

```
mpiexec.exe -p <smpd_port> -hosts <num_of_hosts>
<hosts_ip_list> -env MPICH_NETMASK <network_ip/subnet> -
env MPICH_ND_ZCOPY_THRESHOLD -1 -env MPICH_DISABLE_ND <0/
1> -env MPICH_DISABLE_SOCK <0/1> -affinity <process>
```

B.3 Directing MSMPI Traffic

Directing MPI traffic to a specific QoS priority may delayed due to:

- Except for NetDirectPortMatchCondition, the QoS powershell CmdLet for NetworkDirect traffic does not support port range. Therefore, NetwrokDirect traffic cannot be directed to ports 1-65536.
- The MSMPI directive to control the port range (namely: MPICH_PORT_RANGE 3000,3030) is not working for ND, and MSMPI chose a random port.

B.4 Running MSMPI on the Desired Priority

Step 1. Set the default QoS policy to be the desired priority (Note: this prio should be lossless all the way in the switches*)

Step 2. Set SMB policy to a desired priority only if SMD Traffic running.

Step 3. **[Recommended]** Direct ALL TCP/UDP traffic to a lossy priority by using the “IPProtocol-MatchCondition”.

TCP is being used for MPI control channel (smpd), while UDP is being used for other services such as remote-desktop.

Arista switches forwards the pcp bits (e.g. 802.1p priority within the vlan tag) from ingress to egress to enable any two End-Nodes in the fabric as to maintain the priority along the route.

In this case the packet from the sender goes out with priority X and reaches the far end-node with the same priority X.

The priority should be losslessin the switches

- ***To force MSMPI to work over ND and not over sockets, add the following in mpiexec command:***

```
-env MPICH_DISABLE_ND 0 -env MPICH_DISABLE_SOCK 1
```

B.5 Configuring MPI

Step 1. Configure all the hosts in the cluster with identical PFC (see the PFC example below).

Step 2. Run the WHCK ND based traffic tests to Check PFC (ndrping, ndping, ndrpingpong, ndpingpong).

Step 3. Validate PFC counters, during the run-time of ND tests, with “Mellanox Adapter QoS Counters” in the perfmon.

Step 4. Install the same version of HPC Pack in the entire cluster.
NOTE: Version mismatch in HPC Pack 2012 can cause MPI to hung.

Step 5. Validate the MPI base infrastructure with simple commands, such as `“hostname”`.

B.5.1 PFC Example

In the example below, ND and NDK go to priority 3 that configures no-drop in the switches. The TCP/UDP traffic directs ALL traffic to priority 1.

- Install dcbx.

```
Install-WindowsFeature Data-Center-Bridging
```

- Remove the entire previous settings.

```
Remove-NetQosTrafficClass
Remove-NetQosPolicy -Confirm:$False
```

- Set the DCBX Willing parameter to false as Mellanox drivers do not support this feature

```
Set-NetQosDcbxSetting -Willing 0
```

- Create a Quality of Service (QoS) policy and tag each type of traffic with the relevant priority.
  In this example we used TCP/UDP priority 1, ND/NDK priority 3.

```
New-NetQosPolicy "SMB" -NetDirectPortMatchCondition 445 -PriorityValue8021Action 3
New-NetQosPolicy "DEFAULT" -Default -PriorityValue8021Action 3
New-NetQosPolicy "TCP" -IPProtocolMatchCondition TCP -PriorityValue8021Action1
New-NetQosPolicy "UDP" -IPProtocolMatchCondition UDP -PriorityValue8021Action 1
```

- Enable PFC on priority 3.

```
Enable-NetQosFlowControl 3
```

- Disable Priority Flow Control (PFC) for all other priorities except for 3.

```
Disable-NetQosFlowControl 0,1,2,4,5,6,7
```

- Enable QoS on the relevant interface.

```
Enable-netadapterqos -Name
```

B.5.2 Running MPI Command Examples

- Running MPI pallas test over ND.

```
> mpiexec.exe -p 19020 -hosts 4 11.11.146.101 11.21.147.101
11.21.147.51
11.11.145.101 -env MPICH_NETMASK 11.0.0.0/
255.0.0.0 -env MPICH_ND_ZCOPY_THRESHOLD -1 -env MPICH_DISABLE_ND 0
-env
MPICH_DISABLE_SOCK 1 -affinity c:\\test1.exe
```

- Running MPI pallas test over ETH.

```
> exempiexec.exe -p 19020 -hosts 4 11.11.146.101 11.21.147.101
11.21.147.51
11.11.145.101 -env MPICH_NETMASK 11.0.0.0/
255.0.0.0 -env MPICH_ND_ZCOPY_THRESHOLD -1 -env MPICH_DISABLE_ND 1
-env
MPICH_DISABLE_SOCK 0 -affinity c:\\test1.exe
```